Rockwell Automation

罗克韦尔自动化技术丛书

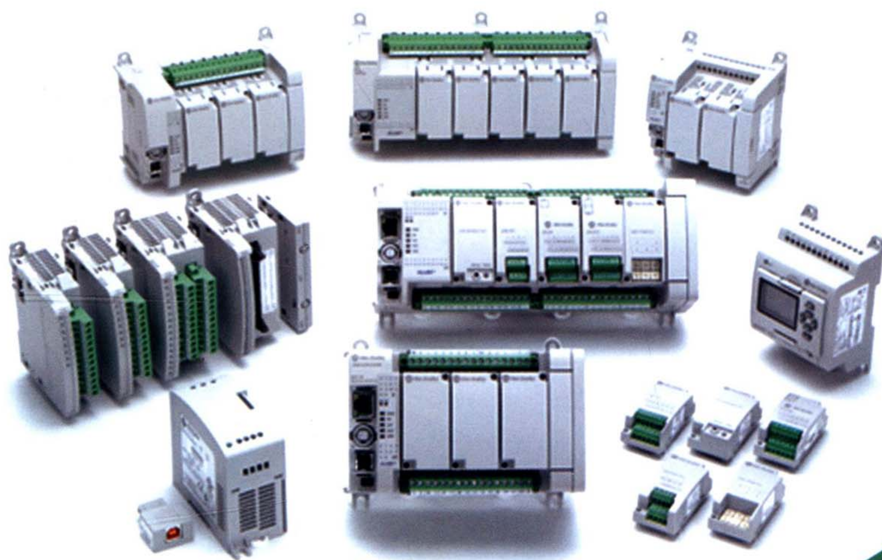
Micro800 控制系统





主　编　钱晓龙　李晓理

副主编　郭　海　李宪英





机械工业出版社

CHINA MACHINE PRESS



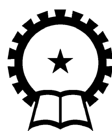
VISIT…

LANZAROTE

Caliente.COM

Micro800 控制系统

主　编　钱晓龙　李晓理
副主编　郭　海　李宪英



机 械 工 业 出 版 社

Micro800 系列控制器是罗克韦尔自动化全新推出的新一代微型 PLC，此系列控制器具有超过 21 种模块化插件，控制器的点数从 10 点到 48 点不等，可以实现高度灵活的硬件配置，在提供足够的控制能力的同时满足用户的基本应用，并且便于安装和维护。不同型号控制器之间的模块化插件可以共用，内置 RS-232、RS-485、USB 和 Ethernet/IP 等通信接口，具有强大的通信功能。免费的编程软件支持功能块一体化编程，并可使用通用的 USB 编程电缆，给编程人员带来了极大的便利；系统还可以提供完整的机器控制方案。Micro800 共有三个系列的控制器，分别为 Micro810、Micro830 和 Micro850。

本书言简意赅、通俗易懂、详细地介绍了 Micro800 系列控制器。通过了解控制器的硬件，理解产品的优势；通过学习编程示例和指令使读者能够熟练地掌握并快速地运用该系列控制器；特别是通过学习速度控制系统的设计，使读者能够更加灵活地使用 Micro800 系列控制器。

本书立足于提高从事自动化专业的工程技术人员和自动化专业的学生对罗克韦尔自动化 Micro800 系列控制器产品的综合运用能力，教会读者如何将 Micro800 系列控制器的功能特点融入工艺当中。本书是结合 Micro800 系列产品的应用类教材，也可作为罗克韦尔自动化的培训教材。

图书在版编目（CIP）数据

Micro800 控制系统/钱晓龙，李晓理主编．—北京：机械工业出版社，2013.1
ISBN 978-7-111-41257-1

Ⅰ.①M…　Ⅱ.①钱…②李…　Ⅲ.①可编程序控制器　Ⅳ.①TP332.3

中国版本图书馆 CIP 数据核字（2013）第 015313 号

机械工业出版社（北京市百万庄大街 22 号　邮政编码 100037）
策划编辑：林春泉　责任编辑：赵　任
版式设计：张　薇　责任校对：陈秀丽
封面设计：鞠　杨　责任印制：张　楠
北京京丰印刷厂印刷
2013 年 2 月第 1 版·第 1 次印刷
184mm×260mm·12.75 印张·310 千字
0 001—3 000 册
标准书号：ISBN 978-7-111-41257-1
定价：34.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务	网络服务
社服务中心：（010）88361066	教材网：http://www.cmpedu.com
销售一部：（010）68326294	机工官网：http://www.cmpbook.com
销售二部：（010）88379649	机工官博：http://weibo.com/cmp1952
读者购书热线：（010）88379203	**封面无防伪标均为盗版**

前　言

自罗克韦尔自动化中国大学项目开始以来，组织出版了大量入门级的教材，以帮助读者尽快掌握罗克韦尔自动化的新产品和新技术。作为大学项目的重要成员——东北大学罗克韦尔自动化实验室自2003年以来出版了十几本教材，主要有Micro系列为主体的小型控制系统系列教材，如《智能电器与MicroLogix控制器》、《MicroLogix控制器应用实例》和《MicroLogix核心控制系统》三本教材。2011年罗克韦尔自动化又推出了Micro800系列控制器，该控制器体积小、功能强、配置灵活、性价比高，非常适用于高校开展实训，更适用于OEM厂商在解决方案上的灵活选择。

本书由东北大学钱晓龙、北京科技大学的李晓理主编。全书共分七章，其中李宪英负责编写第1章Micro810控制器硬件、第2章Micro830控制器硬件，详细地讲述了这两款控制器硬件系统的组成和特点以及可选的嵌入式模块的功能；李晓理负责编写第3章Micro850控制器的硬件，重点介绍了Micro800系列控制器特殊的PTO、HSC功能以及Micro850控制器扩展式模块的特点和使用；钱晓龙负责编写第4章Micro800控制器的网络通信、第7章速度控制系统，讲述了Micro800系列控制器网络通信的几种方法，即RS-232、RS-485、USB和Ethernet/IP等通信方式，以及如何将RS-485组成速度控制系统，内容不仅包括对PowerFlex 4M的使用和控制器的组态，还包括使用触摸屏对系统进行控制；郭海负责编写第5章Micro800控制器的编程示例和第6章Micro800控制器的编程指令，可供编程人员在使用中查阅，并且还介绍了软件的安装、组态等过程。

在本书的编写过程中，得到了有关方面人士的大力支持，书中所有实验均由东北大学罗克韦尔自动化实验室的老师和同学进行了验证，全书涉及的英文资料均由中国传媒大学外国语学院李阳老师负责翻译。本书还得到了新世纪优秀人才支持计划资助（NCET-11-0578）和中央高校基本科研业务费专项资金资助（FRF-TP-12-005B），在这里一并表示感谢。本书是在罗克韦尔自动化大学项目经理李磊先生的积极推动下完成的，罗克韦尔自动化公司OEM产品经理尚德威先生和市场部的胡俊锦先生参与了本书提纲的编写，并提供了大量的素材以及最后的审核工作。罗克韦尔自动化中国大学项目部的李淼和吕颖珊女士也一直关注本书的出版，对本书提出了大量宝贵意见的同时还给予了我们各方面的帮助，在此表示最诚挚的谢意。

全书以Micro800控制器的使用为基调，同时兼顾与PanelView人机界面和PowerFlex4M变频器的系统组成。可以说本书是对Micro800控制系统灵活运用的归纳与总结，本书的着眼点是教会读者如何将应用放在第一位，使Micro800控制器的特点能够在实战中得到淋漓尽致的发挥。

由于编者水平有限，特别是对Micro800控制器的扩展单元在实际应用中的积累还很不够，书中难免有错误和不妥之处，敬请广大读者批评指正。

编者于东北大学

2012年9月29日

目　　录

前言
第 1 章　Micro810 控制器硬件 …… 1
1.1　Micro810 的硬件特性 …… 2
1.2　Micro810 的 I/O 配置 …… 3
1.3　Micro810 控制器的 LCD 功能 …… 4
1.3.1　模式转换功能 …… 5
1.3.2　智能继电器功能 …… 5
1.3.3　I/O 状态 …… 11
1.3.4　高级设置 …… 12
1.3.5　安全 …… 14
1.4　小结 …… 14
习题 …… 15
思考题 …… 15
第 2 章　Micro830 控制器硬件 …… 16
2.1　Micro830 控制器硬件特性 …… 17
2.2　Micro830 控制器的 I/O 配置 …… 18
2.3　Micro800 控制器嵌入式模块 …… 21
2.3.1　模拟量输入模块 2080-IF4 …… 22
2.3.2　模拟量输出模块 2080-OF2 …… 23
2.3.3　两通道热电偶模块 2080-TC2 …… 25
2.3.4　两通道 RTD 模块 2080-RTD2 …… 26
2.3.5　六通道 TrimPot 模拟量输入模块 2080-TRIMPOT6 …… 29
2.3.6　RS-232/485 隔离串口模块 2080-SERIALISOL …… 30
2.4　小结 …… 30
习题 …… 31
思考题 …… 31
第 3 章　Micro850 控制器硬件 …… 32
3.1　Micro850 控制器硬件特性 …… 33
3.2　Micro850 控制器的 I/O 配置 …… 34
3.2.1　脉冲序列输出 …… 35
3.2.2　高速计数器和可编程限位开关 …… 37
3.3　Micro850 控制器扩展式模块 …… 39
3.3.1　模拟量模块 …… 40
3.3.2　四通道 IRT4 模块 …… 43
3.4　添加扩展式 I/O …… 45
3.5　编辑扩展 I/O 组态 …… 47
3.6　删除和更换扩展 I/O 组态 …… 52

3.7　小结 …… 54
习题 …… 54
思考题 …… 54
第4章　Micro800控制器的网络通信 …… 55
4.1　Micro800控制器的网络结构 …… 56
4.2　USB通信 …… 57
4.3　RS-485通信 …… 57
4.4　RS-232串口通信 …… 62
4.5　Ethernet通信 …… 67
4.6　小结 …… 68
习题 …… 68
思考题 …… 69
第5章　Micro800控制器的编程示例 …… 70
5.1　编程软件的安装 …… 71
5.2　编程软件简介 …… 71
5.2.1　创建一个新项目 …… 71
5.2.2　组态控制器嵌入模块 …… 72
5.2.3　创建一个新程序 …… 73
5.2.4　组态变频器 …… 75
5.3　编程示例 …… 83
5.3.1　自定义功能块示例 …… 83
5.3.2　自定义功能块的使用 …… 87
5.3.3　程序的导入和导出 …… 89
5.4　程序的调试和下载 …… 91
5.4.1　程序的下载和上传 …… 91
5.4.2　程序的调试 …… 93
5.5　小结 …… 94
习题 …… 95
思考题 …… 95
第6章　Micro800控制器的编程指令 …… 96
6.1　Micro800控制器编程语言 …… 97
6.1.1　梯形图 …… 97
6.1.2　功能块 …… 98
6.1.3　结构文本 …… 99
6.2　Micro800控制器的内存组织 …… 102
6.2.1　变量 …… 102
6.2.2　程序 …… 103
6.3　Micro800控制器的指令系统 …… 103
6.3.1　梯形图常用指令 …… 103
6.3.2　功能块指令 …… 109
6.3.3　函数指令 …… 154
6.3.4　运算符 …… 168

6.4 小结 …… 172
习题 …… 172
思考题 …… 172
第 7 章 速度控制系统 …… 173
7.1 速度控制系统结构 …… 174
7.2 PowerFlex 4M 交流变频器 …… 175
7.2.1 PowerFlex 4M 变频器选型 …… 176
7.2.2 PowerFlex 4M 的 I/O 端子接线 …… 178
7.2.3 PowerFlex 4M 集成式键盘操作 …… 179
7.2.4 PowerFlex 4M 的 Modbus 网络通信 …… 182
7.3 速度控制系统设计 …… 186
7.4 小结 …… 192
习题 …… 192
思考题 …… 192
附录 …… 193
参考文献 …… 195

第 1 章

Micro810 控制器硬件

学习目标

- 了解 Micro810 控制器的基本功能
- 掌握 Micro810 控制器输入/输出的使用方法
- 掌握 Micro810 控制器硬件的使用方法
- 了解 Micro810 控制器的高级功能

Micro800 系列控制器是罗克韦尔自动化全新推出的新一代微型 PLC，此系列控制器具有超过 21 种模块化的插件，控制器的点数从 10 点到 48 点不等，可以实现高度灵活的硬件配置，在提供足够的控制能力的同时满足用户的基本应用，并且便于安装和维护。不同型号的控制器之间的模块化插件可以共用，内置 RS-232、RS-485、USB 和 Ethernet/IP 等通信接口，有强大的通信功能。免费的编程软件支持功能块一体化编程，并可使用通用的 USB 编程电缆，给编程人员带来了极大的便利；还可以提供完整的机器控制方案。Micro800 共有 3 个系列的控制器，分别为 Micro810、Micro830 和 Micro850，它们型号的含义如图 1-1 所示。

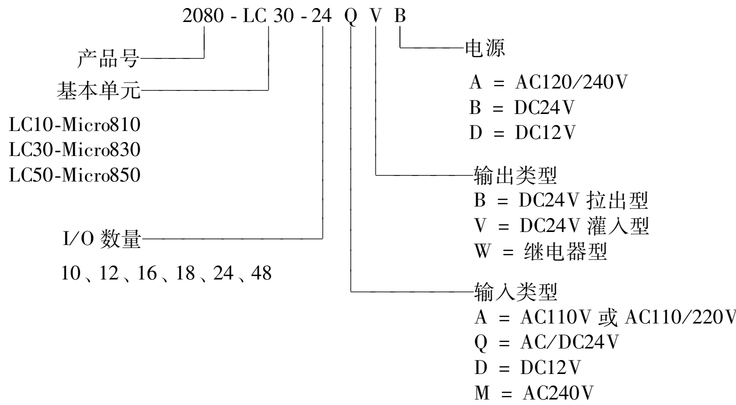


图 1-1 控制器型号含义

1.1 Micro810 的硬件特性

Micro810 控制器按照其 I\O 点数可以分为三种款型:12 点、18 点和 24 点。具体如下：12 点：2080-LC10-12QWB、2080-LC10-12AWA、2080-LC10-12QBB、2080-LC10-12DWD；18 点：2080-LC10-18QWB、2080-LC10-18AWA、2080-LC10-18QBB、2080-LC10-18MWA；24 点：2080-LC10-24QWB、2080-LC10-24AWA、2080-LC10-24QBB、2080-LC10-24 MWA。

12 点控制器的外形如图 1-2 所示，它是一种固定式的控制器，硬件说明见表 1-1。

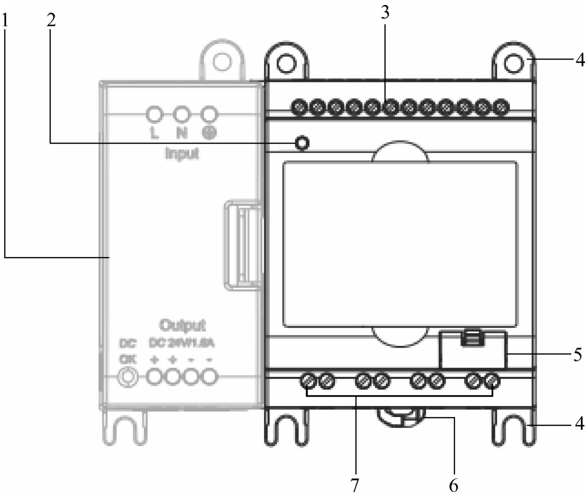


图 1-2 Micro810 控制器外形（12 点）

表 1-1 Micro810（12 点）控制器硬件说明

标号	描　　述
1	电源模块
2	电源指示灯
3	输入接线端子
4	安装孔
5	USB 端口,仅用来插 USB 适配器模块
6	DIN 导轨卡件
7	输出接线端子

1.2　Micro810 的 I/O 配置

Micro810 控制器有 12 种型号，不同型号控制器的 I/O 配置不同。控制器的 I/O 数据见表 1-2。

表 1-2　Micro810 控制器 I/O 数据表

控制器	输入				输出		模拟量输入 0～10V（与直流输入共用）
	AC 120V	AC 240V	DC/AC 24V	DC 12V	继电器	DC24V 拉出型	
2080-LC10-12QWB			8		4		4
2080-LC10-12AWA	8				4		
2080-LC10-12QBB			8			4	4
2080-LC10-12DWD				8	4		4
2080-LC10-18QWB			12		6		4
2080-LC10-18AWA	12				6		
2080-LC10-18QBB			12			6	4
2080-LC10-18MWA		12			6		
2080-LC10-24QWB			16		8		4
2080-LC10-24AWA	16				8		
2080-LC10-24QBB			16			8	4
2080-LC10-24MWA		16			8		

下面以 2080-LC10-12QWB 控制器为例介绍 Micro810 控制器输入/输出端子的使用方法。该控制器的外部接线端子图如图 1-3 所示。

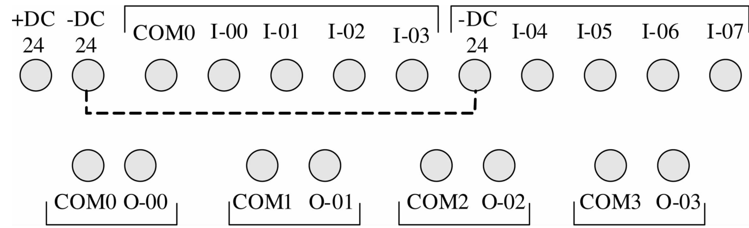


图 1-3　Micro810 控制器外部接线端子图（12 点）

此模块有 4 路数字量输出，都是继电器类型，8 路数字量输入，其中 I-04～I-07 既作为数字量输入，也作为 4 路模拟量输入，它们共用一路端子。

12 点的 Micro810 控制器不能使用 Micro800 控制器其他的嵌入式或者扩展式模块，但是它支持 USB 适配器模块和 LCD 模块，其中 LCD 模块可以作为内存备份模块。

注意：在通电的情况下，嵌入和拔出模块会产生电弧，可能会造成人身伤害或者设备损坏。所以在操作环境不安全的情况下，嵌入和拔出模块之前一定要确保断电，这样才不会因为电弧而造成危害。

1.3 Micro810 控制器的 LCD 功能

2080-LCD 模块可以作为 Micro810 控制器的内存备份模块，它给监视控制器状态和组态控制器带来了便利。在安全的环境下，此模块支持热插拔。

首先来介绍 LCD 模块面板上按钮功能，见表 1-3。

表 1-3 按钮功能

按钮	功能
“OK”键	进入下一级菜单；存储输入信息；应用更改内容
方向键	更改菜单项；更改值；更改位置
“ESC”键	返回上一级菜单；取消自上次保存后所输入的全部内容；多次重复按，返回主菜单

Micro810 控制器的 LCD 面板的操作菜单结构如图 1-4 所示。

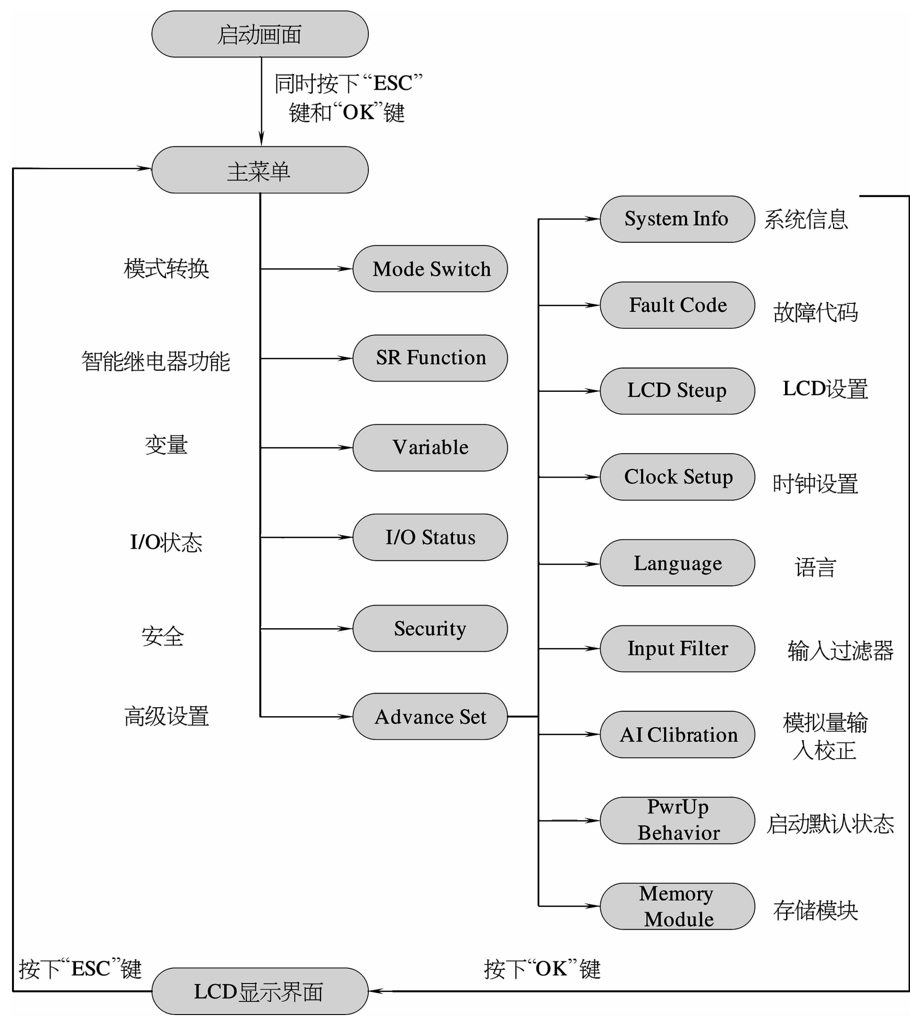


图 1-4 Micro810 控制器的 LCD 面板的操作菜单结构

当控制器上电以后，首先显示的是启动画面，这里默认为 I/O 监视画面，同时按住“OK”键和“ESC”键可以进入主菜单，主菜单有 6 个菜单选项：Mode Switch（模式转换开关）、SR Function（智能继电器功能块）、Variables（变量）、I/O Status（I/O 状态）、Advanced Set（高级设置）和 Security（安全设置）。通过向上/向下箭头来移动光标，对主菜单中各个项的功能进行选择。

1.3.1　模式转换功能

Micro810 控制器的 LCD 模块带有控制器模式转换功能，在主菜单画面，通过 LCD 键盘上的向上或向下键选择 Mode Switch（模式转换开关）。

模式转换开关具有如下位置：PROG Mode（编程模式）和 RUN Mode（运行模式）。可以通过 LCD 上的模式切换画面改变模式转换开关位置，如图 1-5 所示。箭头表明当前模式转换开关的位置。

按向上或向下键，选择所需要的控制器模式，然后按“OK”键将控制器设置成箭头所指的模式。

除了开机信息画面以外的其他 LCD 内置画面也会在右上角显示当前模式转换开关位置，启动画面如图 1-6 所示。本例中，模式转换开关处于 PROG（编程）位置。模式转换完成后，按“ESC”键，返回到主菜单画面。

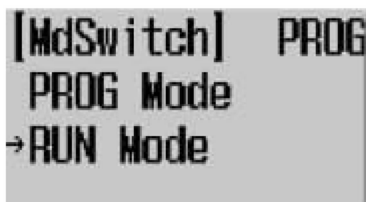


图 1-5　模式转换



图 1-6　启动画面

1.3.2　智能继电器功能

Micro800 控制器的 LCD 面板具有智能继电器功能。在没有电脑的情况下编程人员可以用 LCD 面板对控制器程序进行简单的编写和修改。在主菜单上，通过上下键选择功能块编程，按“OK”进入编程画面。在编程画面的左上角为功能块的编号，用上下键可以改变功能块编号，显示出其他的功能块。选定功能块以后，按左右键可以选择要修改的参数，选定参数后，待修改的参数会闪烁，编程人员可用上下键来修改参数的值，修改完成后按“OK”键进行保存，然后用左右键来选择其他参数。编程画面如图 1-7 所示。

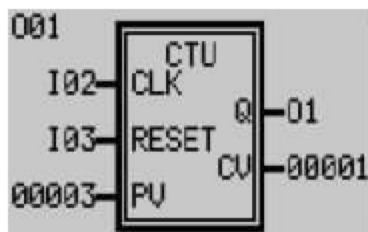


图 1-7　编程画面

12 点的 Micro810 控制器有 8 个智能继电器功能块，编程人员可以使用 LCD 显示器和按钮来组态智能继电器功能块，控制 4 个继电器输出，8 个功能块分别是：

CTU-加计数器；

TON-延时打开计时器；

DOY-当实时时钟的值在年时间设定范围内时，把输

出置为真；

TOW-当实时时钟的值在天时间设定范围内时，把输出置为真；

CTD-减计数器；

TONOF-梯级为真时为延时打开计时器，当梯级为假时为延时断开计时器；

TP-脉冲计时器；

TOF-延时断开计时器。

下面将介绍如何组态上述智能继电器功能块。

1. 组态加计数器（CTU）

1）接通 Micro810 控制器电源。

2）在打开的画面上显示编程模式、时间和 I/O 状态。同时按“ESC”和“OK”键，就会进入主菜单。

3）按向下键，选择智能继电器功能。按“OK”键，显示控制输出 0（即 O0）的功能块。系统先显示控制的输出，选定输出后需要选择使用的功能块。

4）按向上键，这里选择控制输出 1 的功能块。

5）按向右键，选定指令参数，定位到 CTU 功能块。

6）选定现场参数 CLK，这个参数用来触发计数器。

7）选定现场参数 RESET，这个输入参数用来强制计数器复位。

8）设置参数 PV（预设值）为 00003。

9）按向右键，定位到屏幕选择参数。组态 CTU 功能块如图 1-8 所示。

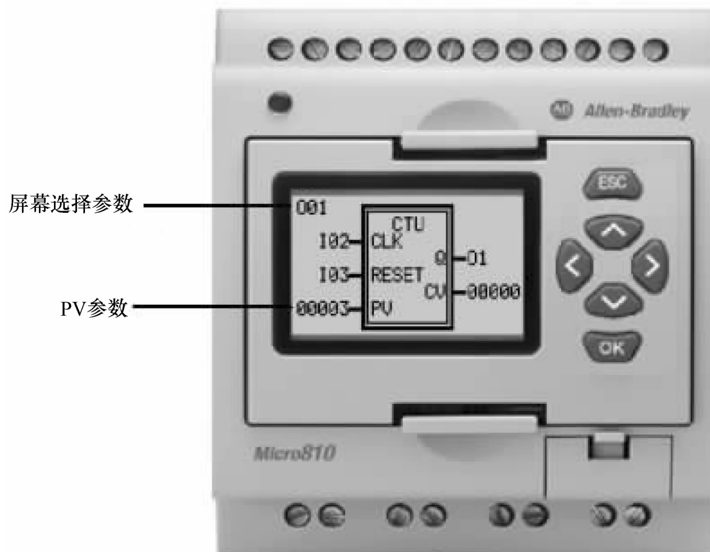


图 1-8　组态 CTU 功能块

10）按“OK”键，保存对参数的修改，此时会弹出一条信息询问是否保存对参数的修改，按“OK”键保存修改。提示信息如图 1-9 所示。

这样就完成了对功能块 CTU 的组态，此组态的预设值是 3，当 CLK 由假到真变化 3 次时，累加值 CV 的值变为 3，和预设值 PV 相等，此时输出位 Q 就会变为真，当按复位键

RESET 时，CV 值被清零。

2. 组态延时打开计时器（TON）

1）接通 Micro810 控制器电源。

2）在打开的画面上显示编程模式、时间和 I/O 状态。同时按“ESC”和“OK”键，就会进入主菜单。

3）按向下键，选择智能继电器功能。按“OK”键，显示控制输出 0 的功能块。

4）按向右键，寻找指令参数，通过上下键找出 TON 指令功能块。

5）按向右键，选择输入参数 IN，这个参数用来启动计时器。

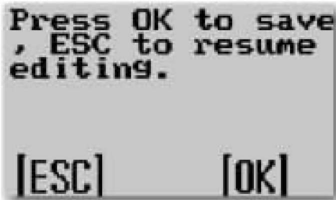


图 1-9　提示信息

6）按向上键，给输入参数 IN 选择适当的输入点。

7）按向右键，选择时间分辨率参数 Time-Resolution，这个参数决定计时器的计时单位。

8）按向下键，改变时间分辨率参数，设为 MM：SS。延时接通计时器组态如图 1-10 所示。

9）按向右键，设置预设值 PT 参数的第一个数字输入，如图 1-10 所示。把 PT 参数的值设定为 02：30。

10）按“OK”键，保存对参数的修改，弹出提示信息后，再次按“OK”键，保存修改。

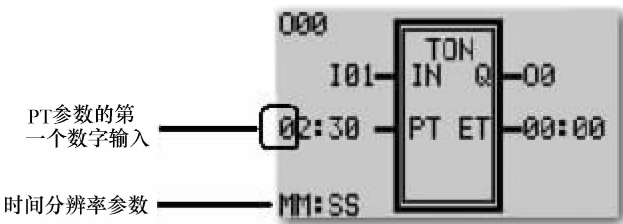


图 1-10　延时接通计时器组态

这样就完成了对 TON 的组态，当输入 IN 由假到真时，计时器开始计时，累加值 ET 的值开始增加，当 ET 的值等于 PT 的值时，输出位 Q 变为真。

3. 组态 DOY 功能块

DOY 功能块的参数设置见表 1-4，所需要的步骤如下：

表 1-4　DOY 功能块参数设置

参数	组　态　值	参数	组　态　值
Q	Q03	Y/C	0
Channel	A	On	11/08/18(YY/MM/DD)
EN	I03	Off	11/08/19(YY/MM/DD)

1）接通 Micro810 控制器电源。

2）在打开的画面上显示编程模式、时间和 I/O 状态。同时按“ESC”和“OK”键，进入主菜单。

3）按向下键，选择智能继电器功能。按“OK”键进入。

4）按向下键，找出控制输出 3 的功能块。

5）按向右键，定位到功能块指令参数。通常默认为 DOY 指令功能块，如果不是可通过上下键定位到 DOY 指令功能块。

6）按向右键，选择通道参数，并显示是通道 A。组态 DOY 功能块如图 1-11 所示。

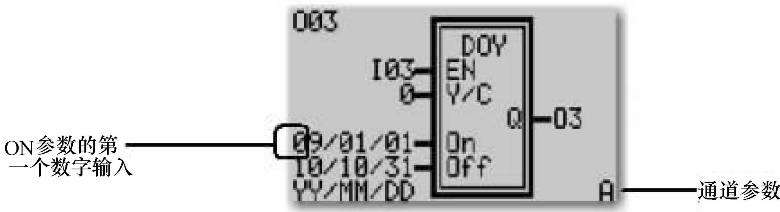


图 1-11　组态 DOY 功能块

7）按向右键，选择 EN 参数。按向下键，把 EN 参数的值变为 I03。

8）按向右键，选择 On 参数的数字输入，如图 1-11 所示，把 On 的日期设定为 11/08/18（YY/MM/DD），完成参数 On 的设置。

9）按向右键，选择 Off 参数的数字输入，用同样方法把它的值设定为 11/08/19（YY/MM/DD）。对 DOY 功能块 On 和 Off 参数的设置如图 1-12 所示。

10）按“OK”键两次保存设置。

DOY 指令的功能是当实时时钟的时间在所设定 4 个通道之中任何一个通道的 On 和 Off 参数之间时，输出位就变为真。所以，当按如上参数设置时，如果实时时钟的时间值位于 2011. 11. 18 和 2011. 11. 19 之间，则输出位将变为真。

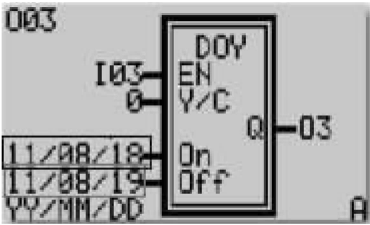


图 1-12　对 DOY 功能块 On 和 Off 参数的设置

4. 组态 TOW 功能块

TOW 功能块的参数设置见表 1-5，所需要的步骤如下：

表 1-5　TOW 功能块参数设置

参数	组　态　值	参数	组　态　值
Q	Q02	D/W	0
Channel	A	On	MO-08:30
EN	I03	Off	MO-08:31

1）接通 Micro810 控制器电源。

2）在打开的画面上显示编程模式、时间和 I/O 状态。同时按“ESC”和“OK”键，就会进入主菜单。

3）按向下键，选择智能继电器功能。按“OK”键，显示控制输出 0 功能块。

4）按向上键，找出控制输出 2 的功能块。

5）按向右键，定位到功能块指令参数。通常默认为 TOW 指令功能块，如果不是可通过上下键定位到 TOW 指令功能块。

6）按向右键，选择通道参数，这里显示为通道 A。

7）按向右键，选择 EN 参数。按向下键，把 EN 参数设置为 I03。

8）按向右键，选择 D/W 参数。

9）按向右键，选择 On 参数的数字输入为 MO-08：30。

10）用同样的方法，把 Off 参数设为 MO-08：31。

11）按“OK”键两次保存修改。组态 TOW 功能块如图 1-13 所示。

TOW 指令的功能是当实时时钟的时间在所设定 4 个通道之中任何一个通道的 On 和 Off 参数之间时，输出位就变为真。所以，当按如上设置参数时，如果实时时钟的时间位于星期一 8：30 到 8：31 之间，则输出位将变为真。

5. 组态减计数器（CTD）

CTD 功能块的参数设置见表 1-6，所需要的步骤如下：

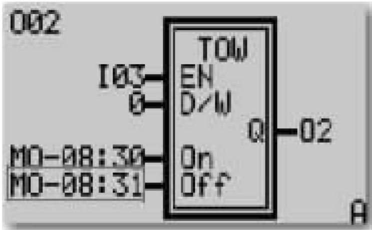


图 1-13　组态 TOW 功能块

表 1-6　CTD 功能块参数设置

参　　数	组　态　值
Q	Q00
CLK	I01
LOAD	I02
PV	00010

1）接通 Micro810 控制器电源。

2）在打开的画面上显示编程模式、时间和 I/O 状态。同时按“ESC”和“OK”键，进入主菜单。

3）按向下键，选择智能继电器功能。按“OK”键，显示控制输出 0 功能块。

4）按向右键，定位功能块指令参数，通过上下键选择 CTD 指令。

5）按向右键，选择 CLK 参数。按向上键，把 CLK 的值设为 I01。

6）按向右键，选择 LOAD 参数。按向上键，选择 I02。

7）按向右键，定位到 PV 数字输入参数，并把值设定为 00010。

8）按“OK”键两次保存修改。组态 CTD 功能块如图 1-14 所示。

指令功能是当 CLK 由假到真变化 10 次时，CV 的值变为 0，此时输出位 Q 就会变为真，但 LOAD 由假到真变化一次后，CV 的值又重新变为 10，输出位 Q 变为假。

6. 组态 TONOF

TONOF 功能块的参数设置见表 1-7，所需要的步骤如下：

1）接通 Micro810 控制器电源。

表 1-7　TONOF 功能块参数设置

参　　数	组　态　值
Q	Q01
IN	I03
Time Resolution	SS：MS
PT	15：000
PTOF	20：000

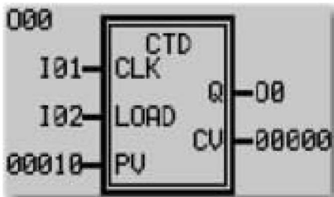


图 1-14　组态 CTD 功能块

2）在打开的画面上显示编程模式、时间和 I/O 状态。同时按“ESC”和“OK”键，进入主菜单。

3）按向下键，选择智能继电器功能。按“OK”键，显示控制输出 0 功能块。

4）按向右键，定位到功能块指令参数，通过上下键选择 TONOF 指令。

5）按向右键，选择参数 IN。这个参数用来启动内部计时器。按向上键，把参数 IN 设为 I03。

6）按向右键，选择 Time Resolution 参数，这个参数决定计时器的计数单元。按向下键，把它设为 SS：MS。

7）按向右键，选择 PT 参数的数字输入，并把值设定为 15：000。

8）按向右键，选择 PTOF 参数，用同样的方法把 PTOF 设置为 20：000。

9）按“OK”键两次保存修改。组态 TONOF 功能块如图 1-15 所示。

TONOF 指令的功能是当输入 IN 由假到真时，ET 开始计时；当 ET 等于 PT 时，输出 Q 变为真（延时通）；同时释放 IN，ET 变为 0。当 IN 再次由假到真时，ET 开始计时，当 ET 等于 PTOF 时，输出 Q 变为假（延时断）。

7. 组态脉冲计时器（TP）

TP 功能块的参数设置见表 1-8，所需要的步骤如下：

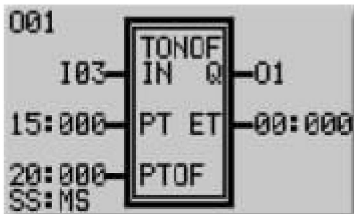


图 1-15　组态 TONOF 功能块

表 1-8　TP 功能块参数设置

参　　数	组　态　值
Q	Q02
IN	I03
Time Resolution	SS：MS
PT	15：000

1）接通 Micro810 控制器电源。

2）在打开的画面上显示编程模式、时间和 I/O 状态。同时按“ESC”和“OK”键，进入主菜单。

3）按向下键，选择智能继电器功能。按“OK”键，显示控制输出 0 的功能块。

4）按向上键，选择控制输出 2 的功能块，按向右键，定位到功能块的指令参数，并用上下键定位到 TP 指令。

5）按向右键，选择参数 IN，这个参数用来启动内部计时器。

6）按向右键，选择 Time Resolution 参数，这个参数决定内部计时器的时间单位。按向下键，设定为 SS：MS。

7）按向右键，选择 PT 参数的数字输入进行设置。

8）按“OK”键两次保存修改。组态 TP 功能块如图 1-16 所示。

此指令完成的功能是当输入 IN 参数有一个由假到真的变化时，启动内部计时器，同时输出 Q 变为真，当累加值 ET 等于预设值 PT 时，输出变为假。

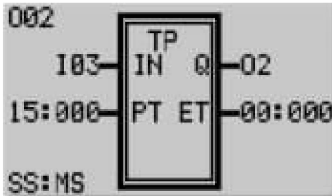


图 1-16　组态 TP 功能块

8. 组态延时断开计时器（TOF）

TOF 功能块的参数设置见表 1-9，所需要的步骤如下：

表 1-9　TOF 功能块参数设置

参　数	组　态　值	参　数	组　态　值
Q	Q03	Time Resolution	SS:MS
IN	I02	PT	15:000

1）接通 Micro810 控制器电源。

2）在打开的画面上显示编程模式、时间和 I/O 状态。同时按“ESC”和“OK”键，进入主菜单。

3）按向下键，选择智能继电器功能。按“OK”键，显示控制输出 0 的功能块。

4）按向下键，选择控制输出 3 的功能块。

5）按向右键，定位到功能块的指令参数，并通过上下键选择 TOF 指令参数。

6）按向右键，选择参数 IN，这个参数用来启动延时断开计时器。按向下键，把参数 IN 的值设定为 I02。

7）按向右键，选择 Time Resolution 参数，这个参数决定内部延时断开计时器的时间单位。按向下键，设定为 SS：MS。

8）按向右键，选择 PT 参数的数字输入进行设置。

9）按“OK”键两次保存修改。组态 TOF 功能块如图 1-17 所示。

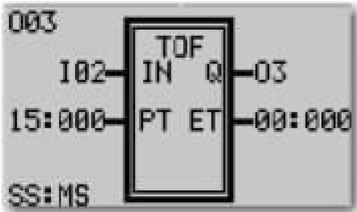


图 1-17　组态 TOF 功能块

此指令的功能是当输入 IN 有一个由假到真的变化时，输出变为真。当输入 IN 由真变假时，延时断开计时器开始计时，当累加值 ET 等于预设值 PT 时，输出就由真变到假。

1.3.3　I/O 状态

Micro810 控制器的 LCD 面板上提供 I/O 状态显示，用于监视控制器和 I/O 状态。在主菜单画面中，使用 LCD 键盘上的向上或向下的键选择 I/O Status（输入/输出状态），然后按 LCD 键盘上的“OK”键，控制器 I/O 状态画面如图 1-18 所示。通常情况下 LCD 的默认画面就是控制器 I/O 状态画面。

图中指示的信息有：

1）通信状态指示器：当矩形闪烁时，说明控制器正通过 USB 接口与 PC 通信。

2）用户显示：当为实心矩形时，表示用户用 LCD 指令编写了一个用户自定义屏幕。

3）密码激活：当密码激活时，显示这个图标；当不激活时，没有显示。

4）操作模式：显示控制器当前的模式，有编程和运行两种模式。

5）控制器时钟时间：显示的是控制器时钟当前的时间。

6）I/O 状态：当输入和输出断电时，显示为空心的矩形，当通电时为实心矩形。

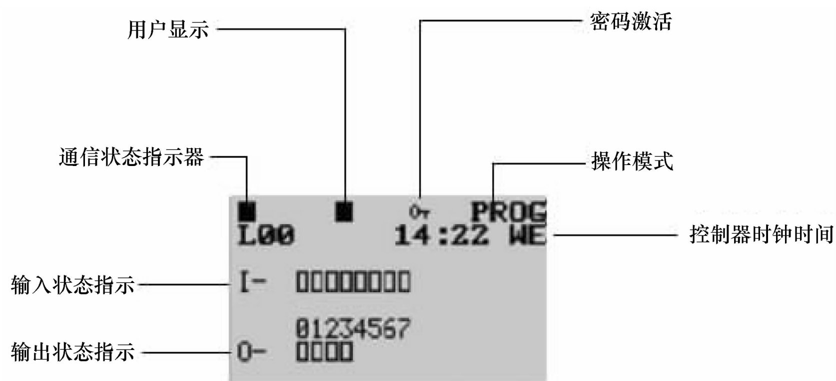


图 1-18　控制器 I/O 状态画面

1.3.4　高级设置

LCD 主菜单下的 Advanced Set（高级设置）子菜单具有以下功能：

- System Info（系统信息）；
- Fault Code（故障代码）；
- LCD Setup（LCD 设置）；
- Clock Setup（时钟设置）；
- Language（语言）；
- Input Filter（输入滤波器）；
- AI Clibration（模拟量输入校正）；
- PwrUp Bhavior（启动默认状态）；
- Memory Module（存储模块）。

选择主菜单画面上的 Advanced Set（高级设置）选项，按“OK”键，进入高级设置菜单画面，如图 1-19 所示。

1. 系统信息

LCD 的系统信息画面允许标识控制器的系统信息。

在 Advanced Set（高级设置）画面中使用 LCD 键盘上的向上或向下键选择 System Info（系统信息），按“OK”键，显示系统信息画面。该画面可以识别控制器的目录号、操作系统固件版本号和开机固件版本号。按“ESC”键返回到高级设置菜单画面。

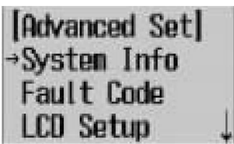


图 1-19　高级设置菜单画面

2. 故障代码

发生故障时，控制器上的 FAULT LED（故障 LED）红色闪烁，由于故障代码画面不会自动显示，因此需要进入故障代码画面查看 LCD 显示的故障代码。

在 Advanced Set（高级设置）画面中使用 LCD 键盘上的向上或向下键选择 Fault Code（故障代码），按“OK”键，显示故障代码画面。如果没有发生故障，显示“0000h”；如果发生故障，显示其故障代码。

按“ESC”键返回到高级设置菜单画面。

3. LCD 设置

在 Advanced Set（高级设置）画面中使用 LCD 键盘上的向上或向下键选择 LCD Setup（LCD 设置）。按“OK”键进入设置画面，这里有 3 个选项：

（1）Contrast（对比度）：按“OK”键进入对比度设置画面，通过左右键可以改变对比度的百分比。

（2）Back Light（背光）：按“OK”键进入背光模式设置画面，这里有三种模式，模式 0 为屏幕一直熄灭，模式 1 为 30s 无操作，屏幕就熄灭，模式 2 为屏幕一直亮。控制器默认为模式 1，通过上下键可以在三种模式之间切换。

（3）P-BUTTON（P 按钮）：按“OK”键进入，有使能和不使能两种状态，通过上下键移动箭头，选择想要设定的状态，按“OK”键即可。

设置完成后，按“ESC”键返回高级设置菜单画面。

4. 时钟设置

Micro810 控制器的 LCD 时钟，可以在 Advanced Set（高级设置）画面中使用 LCD 键盘上的向上或向下键选择 Clock Setup（时钟设置），按“OK”键进入设置画面，选择 CLOCK，按“OK”键进入，这里可以对控制器时钟的年月日时分进行设置。设置完成后，按“ESC”键返回高级设置菜单画面。

5. 语言

Micro810 控制器的 LCD 允许用户给控制器设定所需要的语言，供用户选择的语言有：英语、简体中文、法语、西班牙语和意大利语。在 Advanced Set（高级设置）画面中使用 LCD 键盘上的向上或向下键选择 Language（语言）选项，按“OK”键进入，通过上下键选择所需要的语言，然后按“OK”键保存设置。语言设置选项如图 1-20 所示。设置完成后，按“ESC”键返回高级设置菜单画面。

6. 输入滤波器

Micro810 控制器的 LCD 允许用户给控制器输入设定滤波器，在 Advanced Set（高级设置）画面中使用 LCD 键盘上的向上或向下键选择 Input Filter（输入滤波器）选项，按“OK”键进入，通过上下键选择所需要输入组，然后按“OK”键进入，通过上下键改变对滤波器的设置，按“OK”键保存设置。对滤波器的设置共有以下几种：自定义、28msAC、16msAC、08msAC、32ms、16ms 和 08ms。默认情况下为自定义，对本地输入都要设定为 16ms。设置完成后，按“ESC”键返回高级设置菜单画面。



图 1-20　语言设置选项

7. 模拟量输入校正

Micro810 控制器的 LCD 允许用户对控制器的四组模拟量输入进行校正，在 Advanced Set（高级设置）画面中使用 LCD 键盘上的向上或向下键选择 AI Calibration（模拟量输入校正）选项，按“OK”键进入，通过上下键选择所需要设置的输入组，按“OK”键进入。这里可以设置校正乘数和补偿系数，通过左右键在两个参数之间切换，通过上下键对所选定的值进行改变，默认情况下，校正乘数为 100，补偿系数为 0。设置完成后，按“ESC”键返回高

级设置菜单画面。

8. 启动时的默认状态

Micro810 控制器的 LCD 允许用户对控制器启动时的状态进行设置，在 Advanced Set（高级设置）画面中使用 LCD 键盘上的向上或向下键选择 PwrUp Behavior（启动时的默认状态）选项，按“OK”键进入，通过上下键选择要设定的状态，这里有三种状态，编程模式、运行模式和故障过载。编程模式和运行模式只能选择一个，故障过载可以和其他两个中的一个同时选中。设置完成后，按“ESC”键返回高级设置菜单画面。

9. 存储模块

Micro810 控制器的 LCD 允许用户对存储模块进行设置，在 Advanced Set（高级设置）画面中使用 LCD 键盘上的向上或向下键选择 Memory Module（存储模块）选项，按“OK”键进入，通过上下键选择要设定的项，有以下四种：

（1）MM Setup（存储模块设置）：这里有“总是加载”和“加载打开错误”两种状态。通过上下键，选择所要的状态，按“OK”键即可保持设置。

（2）M800→MM（控制器到存储模块）：把程序由控制器移动到存储模块。当选择此功能时，按“OK”键，会弹出确认信息，要执行此动作，按“OK”键即可。

（3）MM→M800（存储模块到控制器）：把程序由存储模块移动到控制器。当选择此功能时，按“OK”键，会弹出确认信息，要执行此动作，按“OK”键即可。

（4）Clear MM（清除存储模块）：清空存储模块。当选择此功能时，按“OK”键，会弹出确认信息，要执行此动作，按“OK”键即可。

设置完成后，按“ESC”键返回高级设置菜单画面。

1.3.5 安全

Micro810 控制器的 LCD 主菜单的最后一项是安全设置，这里可以对控制器设置一个密码，输入密码后，选择激活密码，控制器就会锁定，用 LCD 对控制器做任何设置都将无法进行，必须先去除密码，才能对控制器进行设置。具体操作如下：

进入主菜单，用向下键定位到“安全”项，按“OK”键，进入密码设置画面，按“OK”键，激活密码，此时在启动界面上就会显示密码激活的图标，如果想要修改对控制器的设置，则要先选择不激活密码，按“OK”键进入，输入密码，再按“OK”键即可。

1.4 小结

本章描述了 Micro810 控制器的特点，并重点介绍了控制器的硬件特性。通过对 Micro810 控制器的组态，掌握和学习了对 Micro810 控制器的使用。由于这种微型的可编程序控制器推向市场及应用时间并不是很长，特别是它还在不断地升级和更新换代，很多功能大家还不是非常熟悉，因此十分需要对它的认识和学习。特别是 Micro810 具有的 LCD 包含了很多功能，编程人员可以在没有计算机的情况下，通过 LCD 编写简单的程序。而且控制器的编程电缆为 USB 接口，方便了项目技术人员使用计算机编程。

习　题

1. Micro810 可编程控制器的有哪些硬件特性?
2. Micro810 可编程控制器支持的电源类型有哪些?
3. Micro810 可编程控制器的输入/输出有几种类型?
4. Micro810 可编程控制器的 LCD 屏包含哪些功能?
5. Micro810 可编程控制器有什么通信端口? 需要辅助通信器件吗?
6. Micro810 可编程控制器可扩展什么类型的模块?
7. Micro810 可编程控制器的 I/O 配置是怎样的?

思考题

1. 请用 Micro810 可编程控制器设计一个简单的交通灯控制系统。
2. 输入滤波的主要目的是什么? 如何计算滤波响应时间?

第 2 章

Micro830 控制器硬件

学习目标

- 了解 Micro830 控制器的基本功能
- 掌握 Micro830 控制器输入/输出的使用方法
- 掌握 Micro830 控制器硬件的使用方法
- 了解 Micro830 控制器的高级功能

2.1　Micro830 控制器硬件特性

Micro830 控制器是带有嵌入式输入/输出的经济型控制器，根据控制器的类型，它可以插 2 ~5 个模块。按照其 I\ O 点数可以分为四种款型：10 点、16 点、24 点和 48 点。具体如下：

10 点：2080-LC30-10QWB、2080-LC30-10QVB；

16 点：2080-LC30-16AWB、2080-LC30-16QWB、2080-LC30-16QVB；

24 点：2080-LC30-24QWB、2080-LC10-24QVB、2080-LC30-24QBB；

48 点：2080-LC30-48QWB、2080-LC30-48AWB、2080-LC30-48QBB、2080-LC10-48QVB。

Micro830 控制器的外形如图 2-1、图 2-2 和图 2-3 所示，它是一种固定式的控制器，具体描述见表 2-1 和表 2-2。

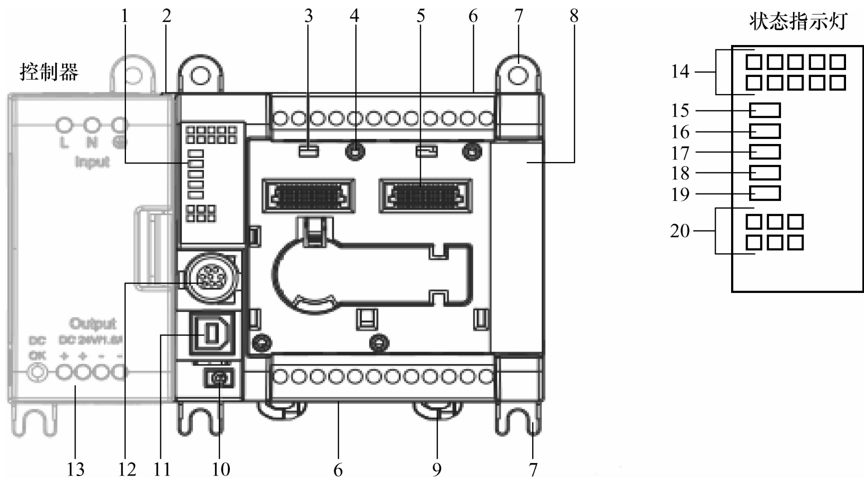


图 2-1　10 \ 16 点 Micro830 控制器和状态指示灯

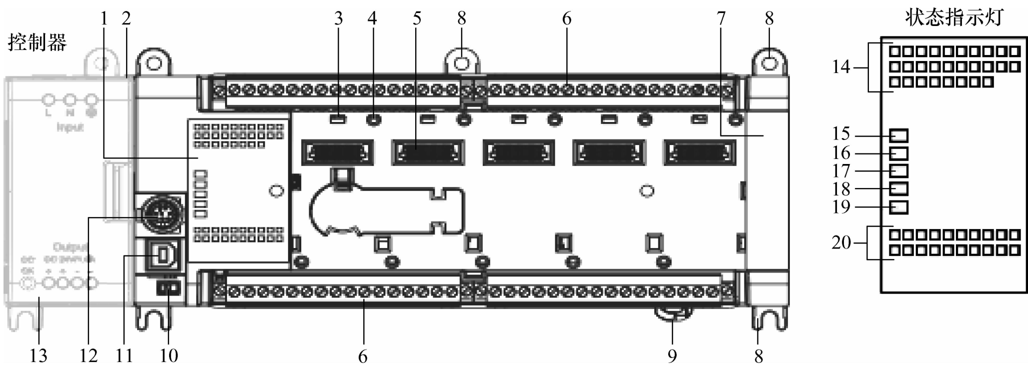


图 2-2　48 点 Micro830 控制器和指示灯

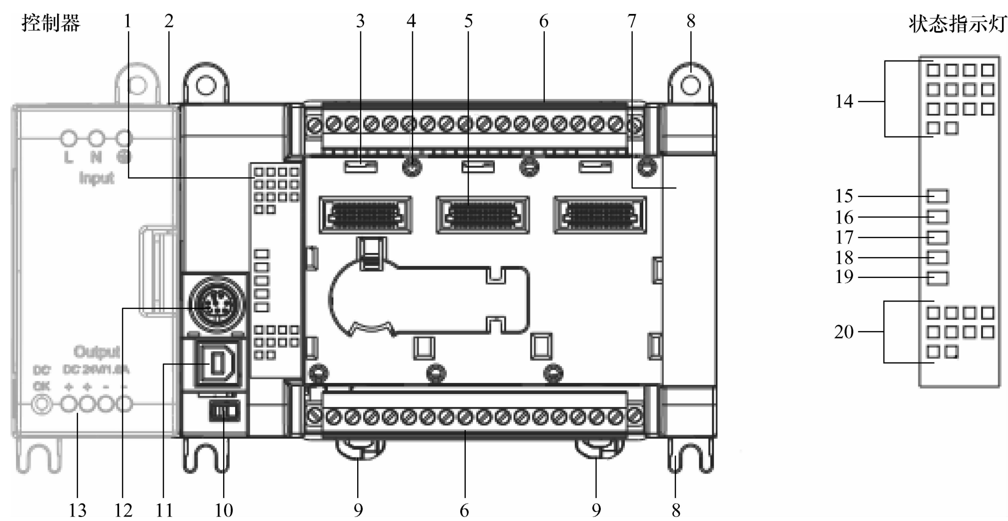


图 2-3　24 点 Micro830 控制器和状态指示灯

表 2-1　Micro830 控制器硬件说明

标号	描　　述	标号	描　　述
1	状态指示灯	8	安装孔
2	电源插槽	9	DIN 导轨安装卡件
3	嵌入式模块卡件	10	模式转换开关
4	嵌入式模块安装孔	11	USB 端口
5	40 针高速插件连接器	12	RS-232/RS-485 通信串口(非隔离)
6	可移动 I/O 接线端子	13	交流电源
7	边缘侧盖		

表 2-2　Micro830 控制器状态指示灯说明

标号	描　　述	标号	描　　述
14	输入指示灯	18	强制 I/O 指示灯
15	电源指示灯	19	串口通信指示灯
16	运行指示灯	20	输出指示灯
17	故障指示灯		

2.2　Micro830 控制器的 I/O 配置

Micro830 控制器有 12 种型号，不同型号的控制器的 I/O 配置不同，控制器的 I/O 数据见表 2-3。

下面以 2080-LC30-24QWB 控制器为例，介绍 Micro830 控制器的输入/输出端子。该控制器的外部接线如图 2-4 所示。其中 I-00 ~ I-07 为高速输入。

表 2-3　Micro830 控制器 I/O 数据

控制器	输　入		输　出		
	AC110V	DC24V/VAC	继电器	24V 灌入型	24V 拉出型
2080-LC30-10QWB		6	4		
2080-LC30-10QVB		6		4	
2080-LC30-16AWB	10		6		
2080-LC30-16QWB		10	6		
2080-LC30-16QVB		10		6	
2080-LC30-24QBB		14			10
2080-LC30-24QVB		14		10	
2080-LC30-24QWB		14	10		
2080-LC30-48AWB	28		20		
2080-LC30-48QBB		28			20
2080-LC30-48QVB		28		20	
2080-LC30-48QWB		28	20		

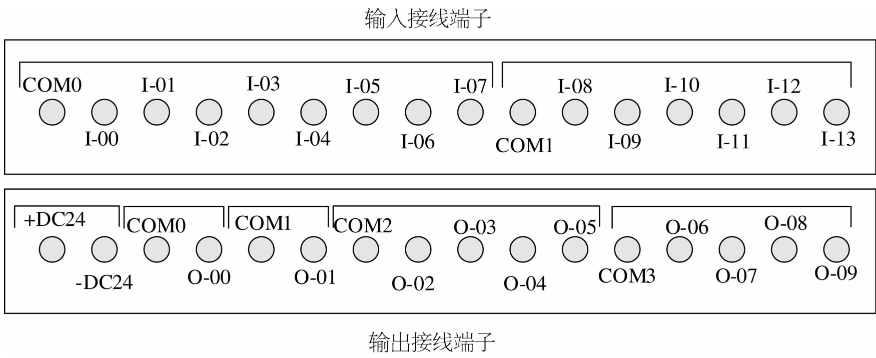


图 2-4　Micro830 控制器外部接线

Micro830 控制器的输入分为灌入型和拉出型，但这仅针对数字量输入，对于模拟量输入则没有灌入型和拉出型之分，其接线图如图 2-5、图 2-6、图 2-7 和图 2-8 所示。

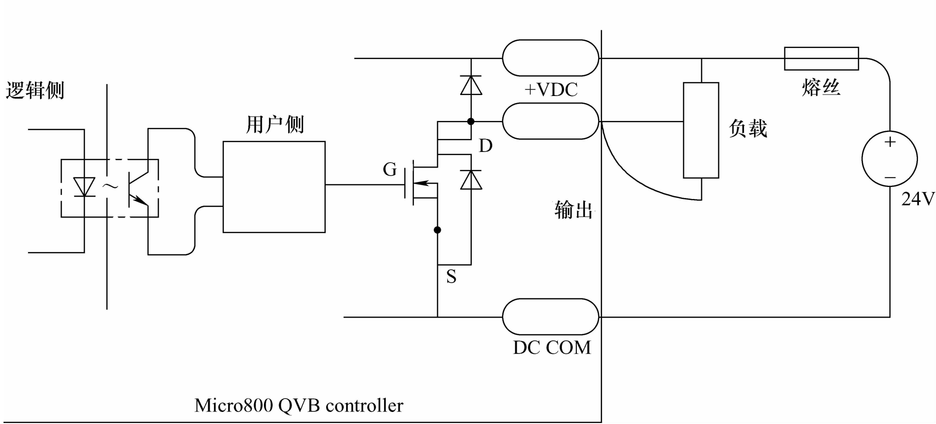


图 2-5　灌入型输出接线图（一）

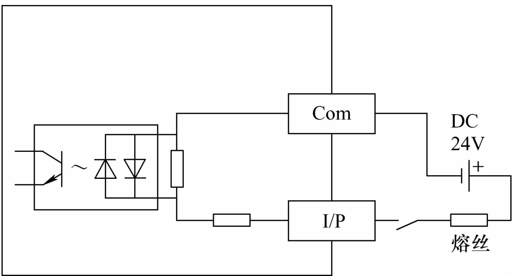


图 2-6　灌入型输入接线图

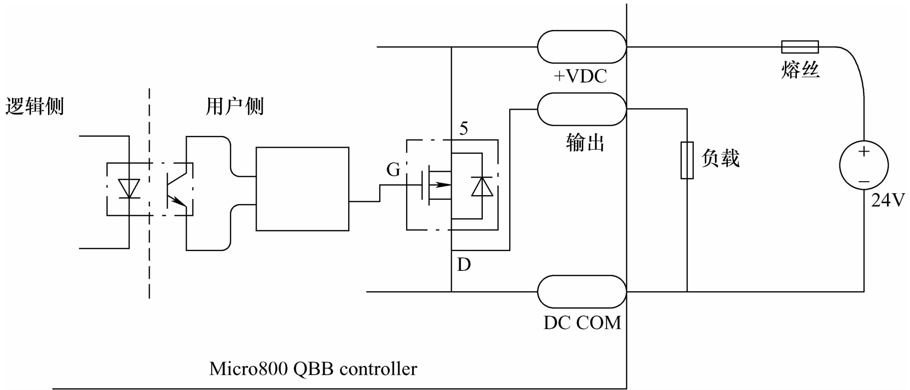


图 2-7　拉出型输出接线图

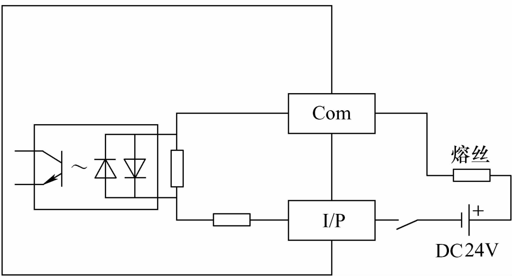


图 2-8　拉出型输入接线图

不同的控制器，高速输入/输出的点不同，具体分布情况见表 2-4。

表 2-4　Micro830 控制器高速输入/输出点的分布情况

控制器型号	高速输入/输出点分布	控制器型号	高速输入/输出点分布
2080-LC30-10QWB	I-00 ~ I03	2080-LC30-24QVB	I-00 ~ I07 、O-00 ~ O01
2080-LC30-10QVB	I-00 ~ I03 、O-00 ~ O01	2080-LC30-24QWB	I-00 ~ I07
2080-LC30-16AWB	没有	2080-LC30-48AWB	没有
2080-LC30-16QWB	I-00 ~ I03	2080-LC30-48QBB	I-00 ~ I11 、O-00 ~ O03
2080-LC30-16QVB	I-00 ~ I03 、O-00 ~ O01	2080-LC30-48QVB	I-00 ~ I11 、O-00 ~ O03
2080-LC30-24QBB	I-00 ~ I07 、O-00 ~ O01	2080-LC30-48QWB	I-00 ~ I11

2.3　Micro800 控制器嵌入式模块

介绍了 Micro830 控制器的硬件特性和本地 I/O 设置以后，下面重点介绍 Micro830 控制器的嵌入式模块，这些模块不仅适用于 Micro830 控制器，还适用于 Micro800 控制器中其他系列的控制器（如 Micro850）。Micro830 控制器最少可以插两个嵌入式模块，最多可以插 5 个嵌入式模块。其嵌入式模块类型见表 2-5。

表 2-5　Micro830 控制器嵌入式模块类型

类型	产品目录号	说　明
数字量 I/O	2080 数字量 I/O	4～8 点 DC24V 数字量 I/O，具有灌入和拉出型-IQ4、OB4、OV4、IQ4OB4、IQ4OV4
	2080-OW4	4 点 1A 继电器输出
模拟量 I/O	2080-IF4	4 通道模拟量输入，0～20mA，0～10V，非隔离，12 位
	2080-IF2	2 通道模拟量输入，0～20mA，0～10V，非隔离，12 位
	2080-OF2	2 通道模拟量输出，0～20mA，0～10V，非隔离，12 位
专用	2080-RTD2	2 通道 RTD，非隔离，0.5C
	2080-TC2	2 通道 TC，非隔离，1C
	2080-TRIMPOT6	6 通道可调电位计模拟量输入，可为速度、位置和温度控制添加 6 个模拟量预设值
	2080-MOT-PTO2	可将基本控制器 PTO 功能扩展至 2 轴（具有 250kHz 差分线路驱动器）
	2080-MOT-HSC2	可将基本控制器 HSC 功能扩展至 2 轴（具有 250kHz 差分线路驱动器）
	2080-MOT-AXIS2	增加 2 轴内插运动，4MHz PTO
	2080-GSM	带 SMS 支持的 GSM 调制解调器
	2080-GSM-DATA	带数据和远程编程能力的 GSM 调制解调器
	2080-MEM-SD	存储器备份-SD 卡适配器
通信	2080-SERIALISOL	RS-232/485 隔离型串行端口
	2080-DNET10	DeviceNet 扫描器主站/从站，用于多达 10 个节点
备份内存	2080-MEMBAK-RTC	高精度实时时钟，备份项目数据和应用项目代码

Micro850 控制器也兼容以上嵌入式模块。了解最新的嵌入式模块信息可以登录到 Rockwell 官方网站查看产品信息。

在介绍 Micro830 控制器的嵌入式模块之前，先来介绍一下控制器的外部电源模块。

在较小的系统中，当没有 DC24V 电源供应时，可以使用型号为 2080-PS120-240VAC 的电源模块，如图 2-9 所示为外部交流供电模块接线图。

其中，PAC-1 为交流电的相线，PAC-2 为交流电的零线，PAC-3 为安全地线；DC-1 和 DC-2 为 +DC24V，DC-3 和 DC-4 为 -DC24V，承受的最大直流电流为 1.6A。

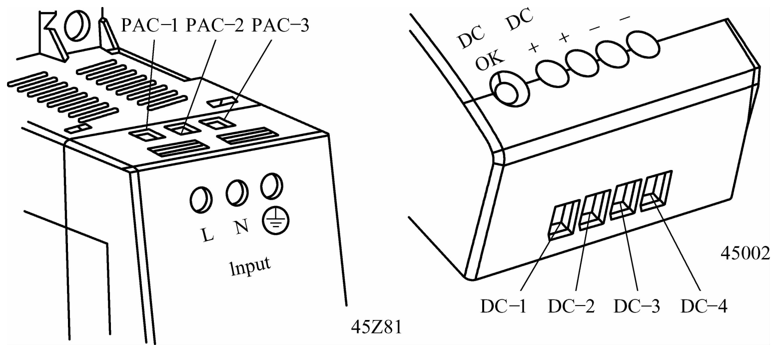


图 2-9 外部交流供电模块接线图

2.3.1 模拟量输入模块 2080-IF4

模拟量输入模块有两通道和四通道两种。嵌入式模块嵌入到 Micro830 控制器的 40 针的嵌入式模块连接器上，嵌入模块后，用固定螺钉固定好，如图 2-10 所示。

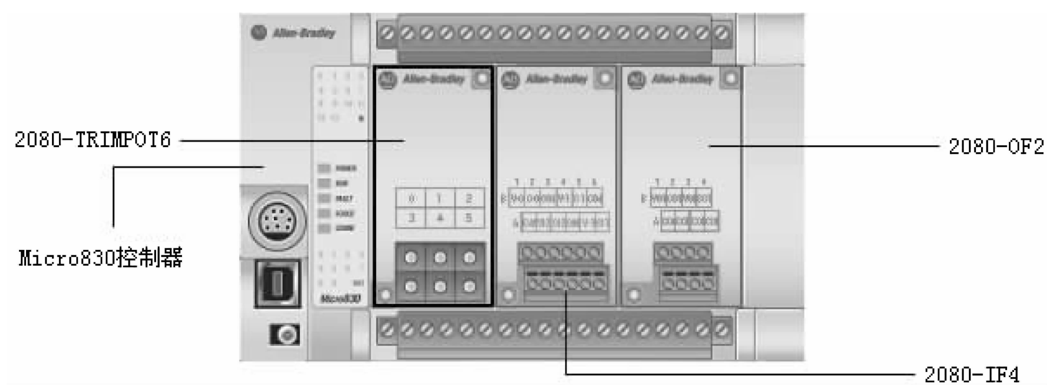


图 2-10 嵌入式模块嵌入到控制器

四通道模拟量输入模块 2080-IF4 的接线端子如图 2-11 所示，各个端子的具体信息见表 2-6。

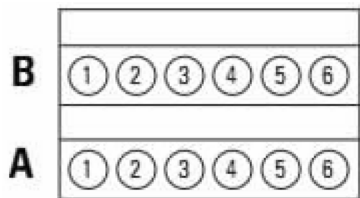


图 2-11 四通道模拟量输入模块接线端子

表 2-6 四通道模拟量输入模块的接线端子信息

端子序号	A	B
1	COM	VI-0(Voltage Input-0)
2	VI-2	CI-0(Current Input-0)
3	CI-2	COM
4	COM	VI-1
5	VI-3	CI-1
6	CI-3	COM

四通道模拟量输入的硬件属性见表 2-7。

表 2-7　四通道模拟量输入的硬件属性

硬件属性	两/四通道模拟量输入模块
模拟量额定工作范围	电压:DC0～10V 电流:0～20mA
最大分辨率	12 位(单极性),在软件中有 50Hz、60Hz、250Hz 和 500Hz
数据范围	0～65535
输入阻抗	电压终端: >220kΩ;电流终端:250Ω
模块误差超过温度范围的百分比(－20～65℃即－4～149 ℉)	电压: ±1.5%;电流: ±2.0%
输入通道组态	通过软件屏幕组态或者通过用户程序组态
输入电路校准	没有要求
扫描时间	180ms
母线隔离的输入组	没有隔离
通道之间隔离	没有隔离
操作温度	－20～65℃即－4～149 ℉
存储温度	－40～85℃(－40～149 ℉)
相对湿度	5%～95%,没有冷凝
操作海拔	2000m
最大电缆长度	10m

四通道模拟量输入模块的接线图如图 2-12 所示。

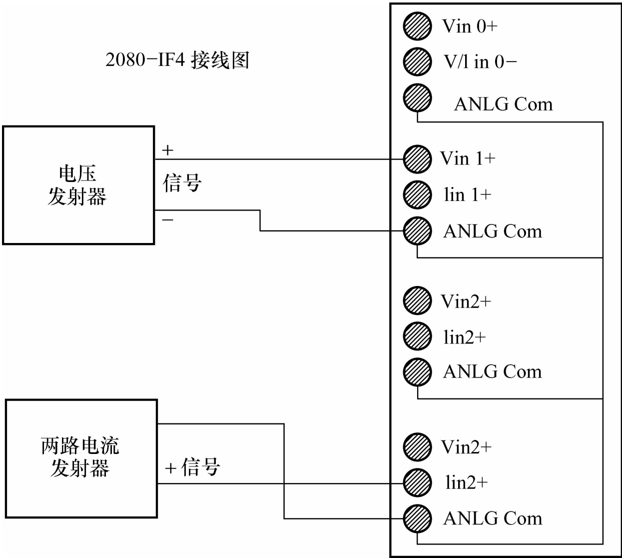


图 2-12　四通道模拟量输入模块的接线图

2.3.2　模拟量输出模块 2080-OF2

2080-OF2 是两通道模拟量输出模块，其接线端子图如图 2-13 所示，各端子的具体信息见表 2-8。

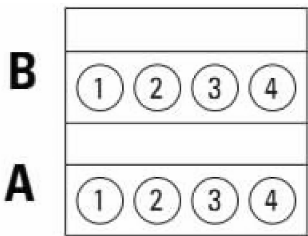


图 2-13　两通道模拟量输出模块接线端子图

表 2-8　两通道模拟量输出模块的接线端子信息

端子序号	A	B
1	COM	V0-0(Voltage Onput-0)
2	COM	C0-0(Current Onput-0)
3	COM	V0-1
4	COM	C0-1

两通道模拟量输出的硬件属性见表 2-9。

表 2-9　两通道模拟量输出硬件属性

硬　件　属　性	两通道模拟量输出模块
模拟量额定工作范围	电压：DC10V；电流：0～20mA
最大分辨率	12 位（单极性）
输出数据范围	0～65535
最大 D/A 转化率（所有通道）	2.5ms
达到 65% 的阶跃响应时间	5ms
电源输出的最大电流负载	10mA
电流输出的电阻负载	0～500Ω
电压输出的负载范围	>1K@DC10V
最大电感性负载（电流输出）	0.01mH
最大电容性负载（电压输出）	0.1μF
输出误差温度范围的百分比（－20～65℃即－4～149 ℉）	电压：±1.5%；电流：±2.0%
开路和短路保护	有
过电压保护	有
母线隔离的输入组	没有隔离
通道之间隔离	没有隔离
操作温度	－20～65℃即－4～149 ℉
存储温度	－40～85℃（－40～149 ℉）
相对湿度	5%～95%，没有冷凝
操作海拔	2000m
最大电缆长度	10m

两通道模拟量输出模块的接线图如图 2-14 所示。

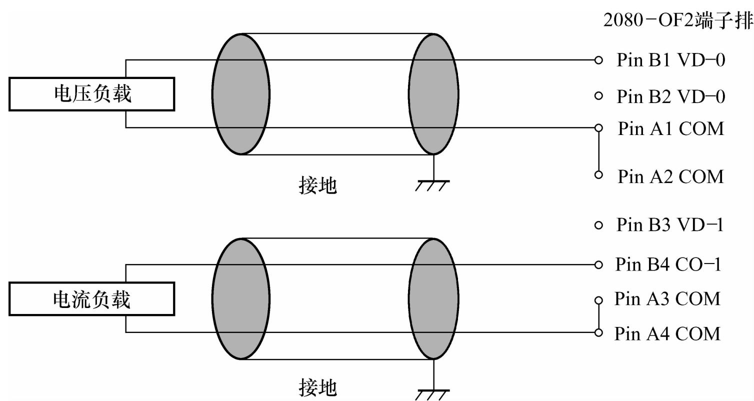


图 2-14　两通道模拟量输出模块的接线图

2. 3. 3　两通道热电偶模块 2080-TC2

2080-TC2 是两通道热电偶模块。它把温度数据传递并转换成数字量信息，它可以接收多达八种温度传感器的信号。可以通过 CCW 软件组态每个单独的通道，组态特定的传感器和滤波频率。

这个模块支持 B、E、J、K、N、R、S 和 T 类型的热电偶传感器。模块的通道称为通道 0、通道 1 和冷端补偿（CJC）。这个冷端补偿由模块外部的 NTC 热敏电阻提供。冷端补偿是通道 0 和通道 1 共有的，提供开路、过载和低于量程的检测和指示。如果通道温度输入低于传感器正常温度范围的最小值，模块就会通过 CCW 全局变量发送一个低于量程的错误。如果通道读数高于最大值，就会发生超量程的报警。规定的热电偶类型和它们相关的温度范围见表 2-10。

表 2-10　规定的热电偶类型和它们相关的温度范围

热电偶传感器类型	温度范围/℃		准确性/℃		ADC 更新速率（准确度℃）
	最小	最大	±1. 0℃	±3. 0℃	
B	40	1820	90 ~ 1700	<90　>1700	4. 17、6. 25、10、16. 7（±1. 0） 19. 6、33、50、62、123、242、470（±3. 0）
E	−270	1000	−200 ~ 930	< −200　>930	
J	−210	1200	−130 ~ 1100	< −130　>1100	
K	−270	1370	−200 ~ 1300	< −200　>1300	
N	−270	1300	−200 ~ 1200	< −200　>1200	
R	−50	1760	40 ~ 1640	<40　>1640	
S	−50	1760	40 ~ 1640	<40　>1640	
T	−270	400	−220 ~ 340	< −220　>340	

该热电偶模块的接线端子如图 2-15 所示。

各端子的具体信息见表 2-11。

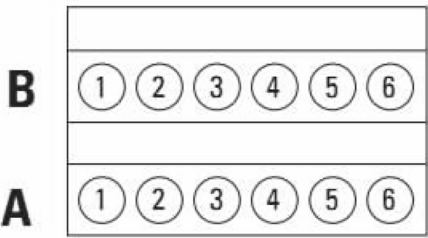


图 2-15　热电偶模块接线端子

表 2-11　两通道热电偶模块的接线端子信息

端子序号	A	B
1	CH0 +	CH1 +
2	CH0 -	CH1 -
3	CJC +	CJC -
4	无连接	无连接
5	无连接	无连接
6	无连接	无连接

图 2-16 为冷端补偿的热敏电阻连接到热电偶模块的接线图，这样连接有助于补偿电压在螺钉连接处的增高，同时热电偶连接到通道 0 和通道 1。

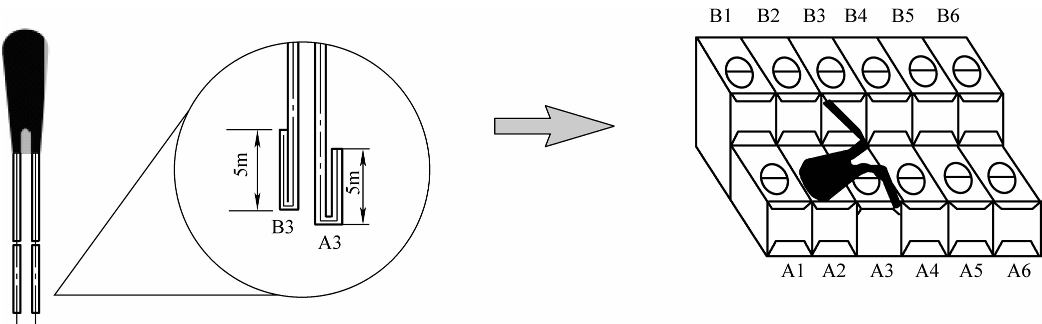


图 2-16　冷端补偿的热敏电阻连接到热电偶模块的接线图

图 2-17 为现场热电偶模块和热电偶传感器的接线图。模块的硬件属性见表 2-14。

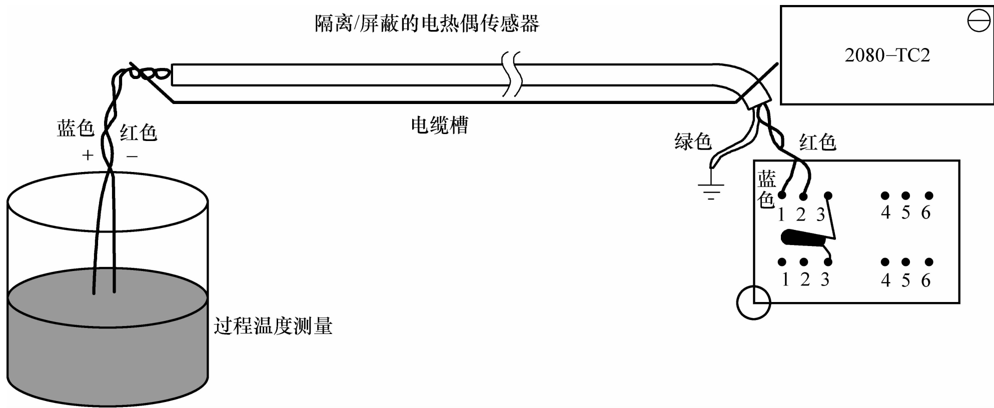


图 2-17　现场热电偶模块和热电偶传感器的接线图

2.3.4　两通道 RTD 模块 2080-RTD2

2080-RTD2 模块支持电阻式温度检测器（RTD）测量，是数字量和模拟量的数据转换模块，并把转换数据传送到它的数据映像表。此模块支持多达 11 种的 RTD。每个通道都在 CCW 软件中单独组态，组态 RTD 的输入后，模块可以把 RTD 读数转换成温度数据。

每个通道提供开路、短路、过载和低于范围的检测和指示。2080-RTD2 模块支持 11 种类型的 RTD。11 种传感器的类型和温度范围见表 2-12。它支持两线和三线类型的 RTD 接线。如果通道温度输入低于传感器正常温度范围的最小值，模块就会通过 CCW 软件全局变量发送一个低于量程的错误。如果通道读数高于最大值，就会发生超量程的报警。

表 2-12　11 种传感器的类型和温度范围

传感器类型	温度范围/℃		准确性/℃		ADC 更新速率/Hz（准确度℃）
	最小	最大	±1.0℃	±3.0℃	
PT100 385	-200	660	-150~590	< -150　>590	三线 4.17、6.25、10、16.7、19.6、33、50（±1.0）62、123、242、470（±3.0） 两线和三线 Cu10 4.17、6.25、10、16.7（>±1.0、<±3.0）19.6、33、50、62、123、242、470（±3.0） 两线 4.17、6.25、10、16.7（±1.0）19.6、33、50、62、123、242、470（±3.0）
PT200 385	-200	630	-150~570	< -150　>570	
PT500 385	-200	630	-150~580	< -150　>580	
PT1000 385	-200	630	-150~570	< -150　>570	
PT100 392	-200	660	-150~590	< -150　>590	
PT200 392	-200	630	-150~570	< -150　>570	
PT500 392	-200	630	-150~580	< -150　>580	
PT1000 392	-50	500	-20~450	< -20　>450	
Cu10 427	-100	260		< -70　>220	
Ni120 672	-80	260	-50~220	< -50　>220	
NiFe604 518	-200	200	-170~170	< -170　>170	

2080-RTD2 模块的接线端子如图 2-18 所示，其接线端子具体信息见表 2-13。

表 2-13　两通道 RTD 模块的接线端子信息

端子序号	A	B
1	CH0 +	CH1 +
2	CH0 -	CH1 -
3	CH0L	CH1L
4	无连接	无连接
5	无连接	无连接
6	无连接	无连接

图 2-18　2080-RTD2 模块的接线端子

图 2-19 为传感器连接图。由于三线和四线 RTD 在嘈杂的工业环境中准确性较好，所以较为常用。当使用这些传感器的时候，长度导致的额外电阻由额外的第三根线（三线制中）或额外的两根线（四线制中）进行补偿。在这个模块的两线制 RTD 中，这些电阻的补偿使用一个额外的 50mm22AWG 的短线提供，这些短线分别连接在通道 0 和通道 1 的 A2、A3 和 B2、B3 上。

图 2-20 为 RTD 模块和 RTD 在现场的连接。RTD 的传感元件应该总是连接在通道 1 的 B1（+）和 B2（-）端子之间，以及通道 0 的 A1（+）和 A2（-）端子之间。B3 和 A3 端子则应该分别短接到 B2 和 A2，来完成电流回路。不正确的接线将导致错误的读数。

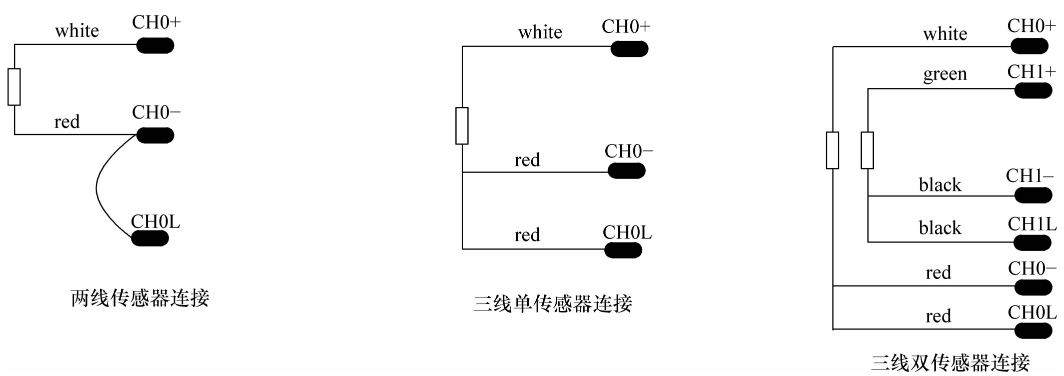


图 2-19　传感器连接图

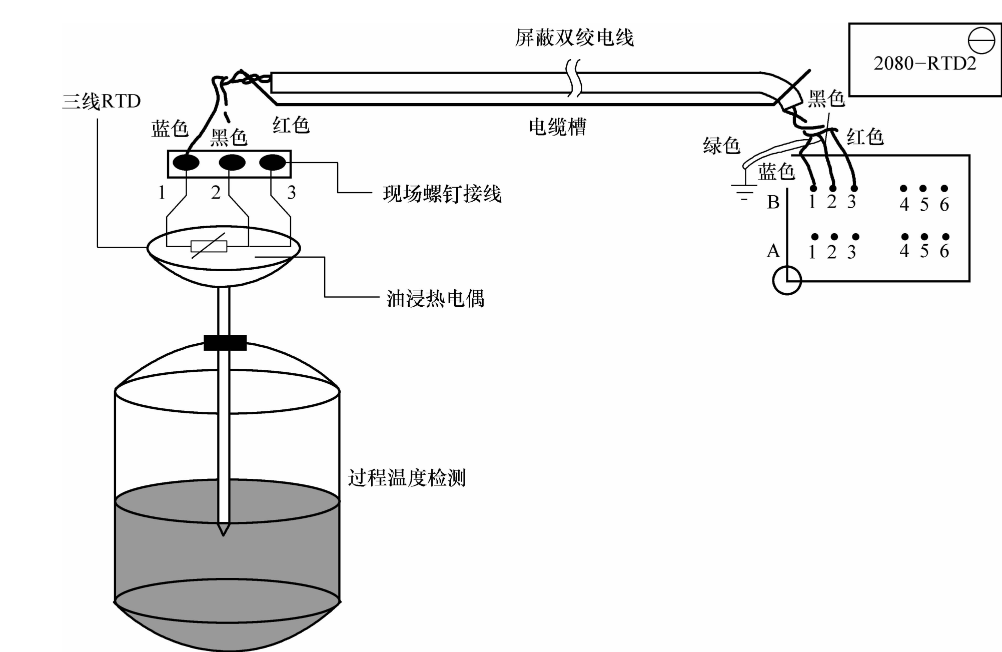


图 2-20　RTD 模块和 RTD 在现场的连接

2080-TC2 和 2080-RTD2 模块的硬件属性见表 2-14。

表 2-14　2080-TC2 和 2080-RTD2 模块硬件属性

属　　性	2080-RTD2	2080-TC2
安装扭矩	0. 2N · m(1. 48lb-in)	
终端螺杆扭矩	0. 22 ~ 0. 25N · m(1. 95 ~ 2. 21li-in) 使用 2. 5mm 的平口螺钉旋具	
线型	0. 14 ~ 1. 5mm^2(26 ~ 16AWG)刚性铜线或者 0. 14 ~ 1. 0mm^2 (26 ~ 17AWG)刚性铜线	
输入阻抗	> 5MΩ	> 300kΩ
共模抑制比	100dB 50/60Hz	
常模抑制比	70dB 50/60Hz	

（续）

属　　性	2080-RTD2	2080-TC2
分辨率	14 位	
CJC 错误	—	±1.2℃@25℃
准确性	±1.0℃@25℃	
通道	2 非隔离	
短路检测时间	8～1212ms	8～1515ms
功率消耗	3.3V 40mA	
环境空气最大温度	65℃	
操作温度	-20～65℃	
存储温度	-40～85℃	

2080-TC2 和 2080-RTD2 模块帮助用户通过 PID 实现温度自动控制。这两个嵌入模块可以插在 Micro830 控制器的任意槽，但是不支持带电插拔。

2.3.5　六通道 TrimPot 模拟量输入模块 2080-TRIMPOT6

2080-TRIMPOT6 模块提供了一种对速度、位置和温度的控制，可增加六种模拟量预置。该嵌入模块可以插在 Micro830 控制器的任意槽，但是不支持带电插拔。模块的外观和端子图如图 2-21 所示。

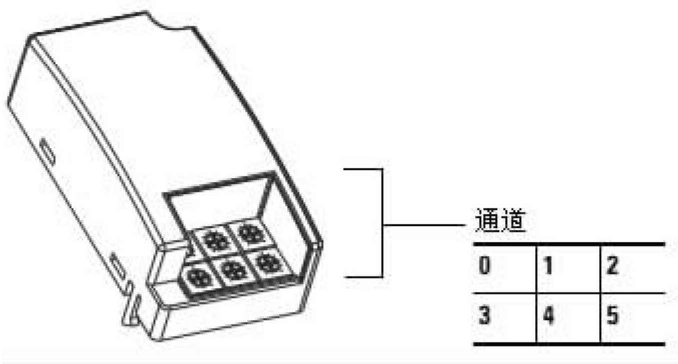


图 2-21　2080-TRIMPOT6 模块的外观和端子图

2080-TRIMPOT6 模块的属性见表 2-15。

表 2-15　2080-TRIMPOT6 模块的属性

硬件属性	数　　值	硬件属性	数　　值
数据范围	0～255	相对湿度	5%～95%，没有冷凝
操作温度	-20～65℃即 -4～149 ℉	操作海拔	2000m
存储温度	-40～85℃（-40～149 ℉）		

2.3.6 RS-232/485 隔离串口模块 2080-SERIALISOL

2080-SERIALISOL 模块支持 RTU 和 ASCII 协议。这个端口是电气隔离的，所以是用在嘈杂设备上的理想选择，例如变频器和伺服设备，并且通信电缆较长，在采用 RS-485 时长度可达 100m。图 2-22 为该模块的接线端子图，端子的具体信息见表 2-16。

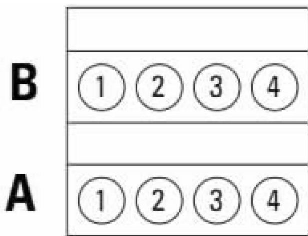


图 2-22 2080-SERIALISOL 模块的接线端子图

表 2-16 2080-SERIALISOL 模块的接线端子信息

端子序号	A	B
1	RS-485 +	RS-232 DCD
2	RS-485/232 GND	RS-232 RXD
3	RS-232 RTS	RS-232 TXD
4	RS-232 CTS	RS-485-

图 2-23 为用 2080-SERIALISOL 模块时的通信线路图。

注意：不能短接 A1 和 B4，如果短接这两个端子，将损坏通信端口。

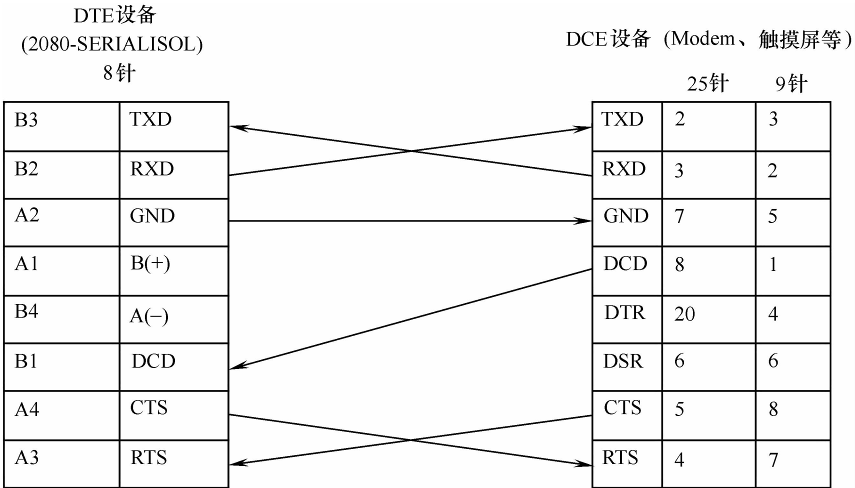


图 2-23 通信线路图

2.4 小结

本章描述了 Micro830 控制器的特点，并重点介绍了控制器的硬件特性。通过对 Micro830 控制器的组态，学习和掌握对 Micro830 控制器的使用。由于这种微型的可编程序控制器推向市场及应用时间并不是很长，特别是它还在不断的升级和更新换代中，很多功能大家还不是非常熟悉，因此十分需要对它的认识和学习。Micro830 控制器的综合能力极强，它不仅将 I/O 点设在处理器上，并且有高速计数模块、高速脉冲输出模块和两轴运动控制嵌入式模块可供嵌入使用。并且该控制器编程采用 USB 电缆，方便编程人员的使用，控制器除了本身具有串行通信端口，支持 MODBUS 和 ASCII 码通信协议以外，还有串行通信模块可

供扩展，这使得网络通信功能更具有灵活性。可以说 Micro800 控制器的功能正变得越来越强大，应用领域也会更加广泛。

习　题

1. Micro830 控制器有哪些硬件特性?
2. Micro830 控制器支持的电源类型有哪些?
3. Micro830 控制器的输入/输出有几种类型?
4. Micro830 控制器的本地 I/O 有哪些功能?
5. Micro830 控制器有几种通信端口?
6. Micro830 控制器可嵌入什么类型的模块? 最多能嵌入几个?
7. Micro830 控制器支持哪些嵌入式模块?
8. Micro830 控制器的 I/O 配置是怎样的?

思考题

1. 请用 Micro830 控制器设计一个简单的交通灯控制系统，对比用 Micro810 控制器设计的系统，说说各自的特点。

2. 比较 Micro830 和 Micro810 控制器，说说各自的优缺点。

第 3 章

Micro850 控制器硬件

学习目标

- 了解 Micro850 控制器的基本功能
- 掌握 Micro850 控制器输入/输出的使用方法
- 掌握 Micro850 控制器硬件的使用方法
- 了解 Micro850 控制器的高级功能

3.1　Micro850 控制器硬件特性

Micro850 控制器是一种可以内置嵌入式 I/O 模块，又可以外挂扩展 I/O 模块的经济型控制器。Micro850 控制器可以嵌入 2 ~5 个模块不等，并且最多能支持 4 个扩展 I/O 模块。

该控制器还可以容纳任何一类 2 级额定 24V 直流输出电源，如符合最低规格的可选 Micro800 电源。按照其 I/O 点数可分为两种款型：24 点和 48 点。

Micro850 控制器外观和状态指示灯如图 3-1、图 3-2 所示，各部分具体描述见表 3-1、表 3-2。

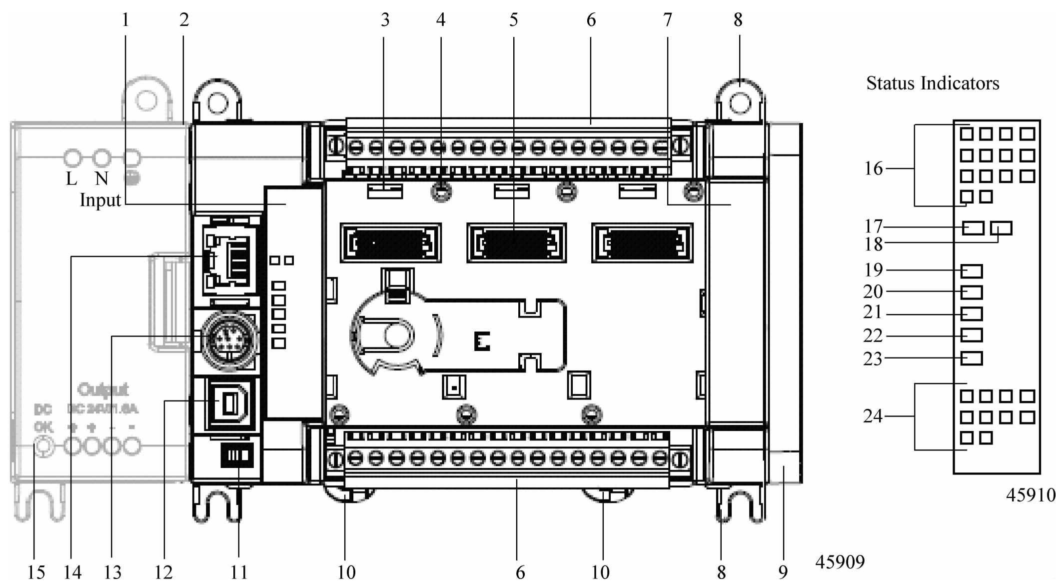


图 3-1　24 点 Micro850 控制器外观和状态指示灯

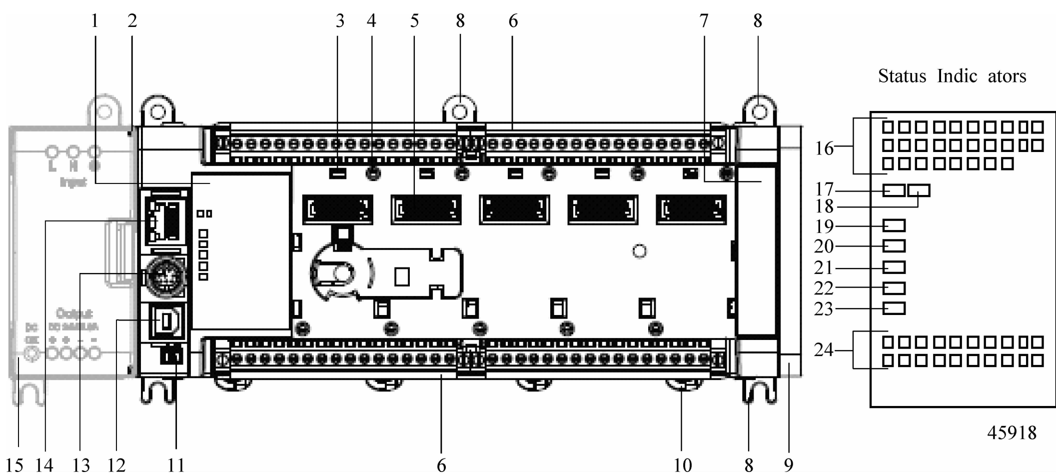


图 3-2　48 点 Micro850 控制器外观和状态指示灯

表 3-1　Micro850 控制器硬件说明

标号	描　　述	标号	描　　述
1	状态指示灯	9	扩展 I/O 槽盖
2	电源插槽	10	DIN 导轨安装卡件
3	嵌入式模块卡件	11	模式转换开关
4	嵌入式模块安装口	12	B 类 USB 端口
5	40 针高速插件连接器	13	RS-232/RS-485 通信串口(非隔离)
6	可移动 I/O 接线端子	14	RJ-45 以太网端口
7	边缘侧盖	15	交流电源
8	安装口		

表 3-2　Micro850 控制器状态指示灯说明

标号	描　　述	标号	描　　述
16	输入指示灯	21	故障指示灯
17	模块指示灯	22	强制 I/O 指示灯
18	网络指示灯	23	串行通信指示灯
19	电源指示灯	24	输出指示灯
20	运行指示灯		

3.2　Micro850 控制器的 I/O 配置

Micro850 控制器有 8 种型号，不同型号的控制器的 I/O 配置不同，控制器的 I/O 数据见表 3-3。

表 3-3　控制器的 I/O 数据

控　制　器	输　　入		输　　出		
	AC120V	$24V_{DC}/V_{AC}$	继电器型	24V 灌入型	24V 拉出型
2080-LC50-24AWB	14		10		
2080-LC50-24QBB		14			10
2080-LC50-24QVB		14		10	
2080-LC50-24QWB		14	10		
2080-LC50-48AWB	28		20		
2080-LC50-48QBB		28			20
2080-LC50-48QVB		28		20	
2080-LC50-48QWB		28	20		

下面以 2080-LC50-24QWB 控制器为例，介绍 Micro850 控制器的输入/输出端子。该控制器的外部接线图如图 3-3 所示。第一排 I-00 ~ I-13 为输入端口，第二排 O-00 ~ O-09 为输出端口。其中，I-00 ~ I-07 为高速输入端口。

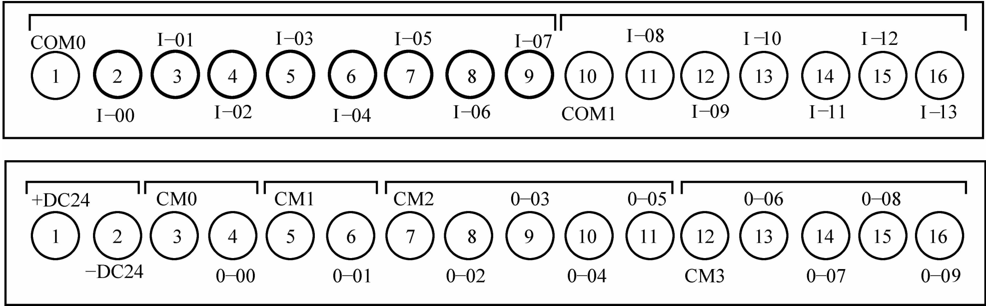


图 3-3　Micro850 控制器外部接线图

Micro850 控制器的输入分为灌入型和拉出型，但这仅针对直流输入而言，并不适用于交流输入。其接线方式与 Micro830 的本地 I/O 接线方式一致，此处不再赘述。

不同型号的控制器，高速输入/输出的点不同，具体分布见表 3-4。

表 3-4　Micro850 控制器高速输入/输出点的分布情况

控制器型号	高速输入/输出点分布	控制器型号	高速输入/输出点分布
2080-LC50-24AWB	I-00 ~ I-07	2080-LC50-48AWB	I-00 ~ I-11
2080-LC50-24QWB	I-00 ~ I-07	2080-LC50-48QWB	I-00 ~ I-11
2080-LC50-24QBB	I-00 ~ I-07、O-00 ~ O-01	2080-LC50-48QVB	I-00 ~ I11、O-00 ~ O03
2080-LC50-24QVB	I-00 ~ I-07、O-00 ~ O-01	2080-LC50-48QBB	I-00 ~ I11、O-00 ~ O03

对于 Micro830 和 Micro850 的特殊本地 I/O 介绍如下。

3.2.1　脉冲序列输出

表 3-5 列出 Micro830 和 Micro850 控制器支持高速脉冲序列输出（Pulse Train Outputs，PTO）数量和运动轴的数量。根据控制器的性能，PTO 功能能够以一个指定频率产生指定数量的脉冲，这些脉冲可以输出到运动控制设备以控制这些设备（如伺服驱动器），进而控制伺服电动机的旋转和位置。每一个 PTO 对应着一个轴，所以可以使用 PTO 功能建立一个位置控制系统。

表 3-5　Micro830 和 Micro850 支持 PTO 和运动轴数量

控制器型号	PTO（本地 I/O）	支持的运动轴
10/16 点 2080-LC30-10QVB；2080-LC30-16QVB	1	1
24 点 2080-LC30-24QVB；2080-LC30-24QBB 2080-LC50-24QVB；2080-LC50-24QBB	2	2
48 点 2080-LC30-48QVB；2080-LC30-48QBB 2080-LC50-48QVB；2080-LC50-48QBB	3	3

注意，对于 Micro830 系列控制器，固件版本需要 2.0 以上才能支持 PTO 功能。

每一个运动轴都需要多个输入/输出信号来控制，Micro830 和 Micro850 控制器带的 PTO 脉冲信号和 PTO 方向信号就可以用来控制轴运动，剩下 PTO 的输入/输出通道可以禁止或是作为普通 I/O 使用。本地 PTO 输入/输出点信息见表 3-6。

表 3-6　本地 PTO 输入/输出点信息

运动控制信号	PTO0(EM _ 00)		PTO1(EM _ 01)		PTO2(EM _ 02)	
	在软件中的名称	在本地端子排名称	在软件中的名称	在本地端子排名称	在软件中的名称	在本地端子排名称
PTO pulse	_ IO _ EM _ DO _ 00	O-00	_ IO _ EM _ DO _ 01	O-01	_ IO _ EM _ DO _ 02	O-02
PTO direction	_ IO _ EM _ DO _ 03	O-03	_ IO _ EM _ DO _ 04	O-04	_ IO _ EM _ DO _ 05	O-05
Lower (Negative) Limit switch	_ IO _ EM _ DI _ 00	I-00	_ IO _ EM _ DI _ 04	I-04	_ IO _ EM _ DI _ 08	I-08
Upper (Positive) Limit switch	_ IO _ EM _ DI _ 01	I-01	_ IO _ EM _ DI _ 05	I-05	_ IO _ EM _ DI _ 09	I-09
Absolute Home switch	_ IO _ EM _ DI _ 02	I-02	_ IO _ EM _ DI _ 06	I-06	_ IO _ EM _ DI _ 10	I-10
Touch Probe Input switch	_ IO _ EM _ DI _ 03	I-03	_ IO _ EM _ DI _ 07	I-07	_ IO _ EM _ DI _ 11	I-11

下面以 2080-LC30-xxQBB/2080-LC50-xxQBB 控制器为例，介绍运动控制系统的搭建。如图 3-4 所示是运动控制系统接线示例。

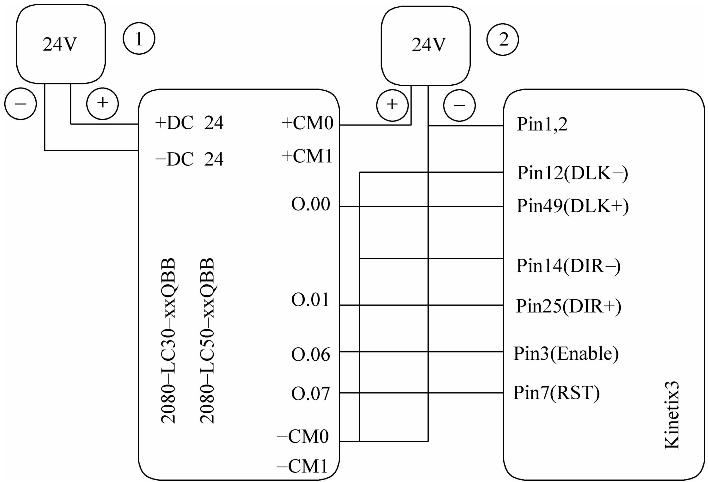


图 3-4　运动控制系统接线示例

注意：如果引脚 1、2 连接的是 24V 电源的正极，使能位（引脚 3）和重置位（引脚 7）

将会变成拉出型输入。

用于搭建运动控制系统的运动控制功能块见表 3-7、3-8。

表 3-7　命令类功能块

功能块名称	功能块名称	功能块名称	功能块名称
MC _ Power	MC _ ReadAxisError	MC _ AbortTrigger	MC _ WriteParameter
MC _ Reset	MC _ ReadParameter	MC _ ReadStatus	MC _ WriteBoolParameter
MC _ TouchProbe	MC _ ReadBoolParameter	MC _ SetPosition	

表 3-8　动作方式类功能块

功能块名称	描　　述	允许执行功能块的运动轴状态
MC _ MoveAbsolute	命令一个轴到指定的绝对值位置	停止，断续运动，连续运动
MC _ MoveRelative	命令一个轴在指定的时间内，运动到一个相对于实际位置指定距离的位置，即在实际位置的基础上再运动指定的距离	停止，断续运动，连续运动
MC _ MoveVelocity	命令轴在指定的速度下，持续运动	停止，断续运动，连续运动
MC _ Home	命令轴执行“search home”序列。当检测到参考信号时，“Position”输入用于设置绝对位置	停止
MC _ Stop	停止轴运动，并使轴处于“Stopping”状态。当轴的速度降为零后，“Done”输出位置 1，轴的状态变为“StandStill”	停止，断续运动，连续运动，回归原点
MC _ Halt	使轴处于一个可控的运动停止状态。轴进入“DiscreteMotion”状态，直至速度降为 0，然后轴变为“StandStill”状态	停止，断续运动，连续运动

3.2.2　高速计数器和可编程限位开关

所有的 Micro830 和 Micro850 控制器都支持高速计数器（High-Speed Counter，HSC）功能，最多的能支持 6 个 HSC。高速计数器功能块包含两部分：一部分是位于控制器上的本地 I/O 端子，另一部分是 HSC 功能块指令（见 6.3.2 节，Input/Output 部分 HSC 功能块指令）。HSC 的参数设置以及数据更新都需要在 HSC 功能块指令中设置。

可编程限位开关（Programmable Limit Switch，PLS）功能允许用户组态 HSC 为 PLS 或者是凸轮开关。

图 3-5 是 HSC 组态为 PLS 的示意图，通过对 HSC 数据结构的设置，可以将 HSC 组态为 PLS 使用。

下面介绍 HSC 的硬件信息：

所有的 Micro830 和 Micro850 控制器，除了 2080-LCxx-xxAWB，都有 100 kHz 的高速计数器。每个主高速计数器有 4 个专用的输入，每个副高速计数器有两个专用的输入。不同点数控制器 HSC 个数见表 3-9。

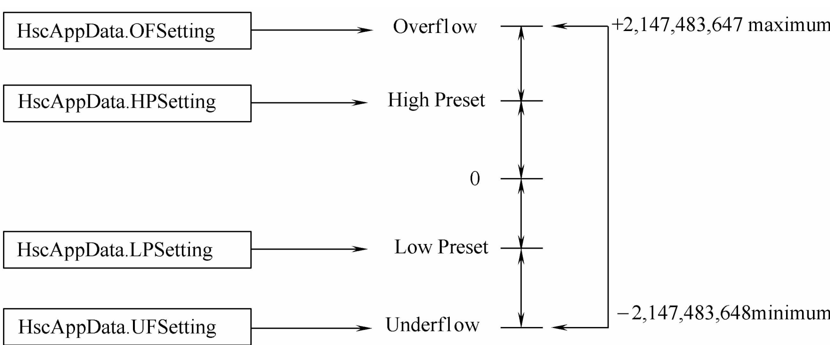


图 3-5　HSC 组态为 PLS 的示意图

表 3-9　不同点数控制器 HSC 个数

	10/16 点	24 点	48 点
HSC 个数	2	4	6
主 HSC	1(counter 0)	2(counter 0,2)	3(counters 0, 2 and 4)
副 HSC	1(counter 0)	2(counter 1,3)	3(counters 1, 3 and 5)

每个 HSC 使用的本地输入号见表 3-10。

表 3-10　每个 HSC 使用的本地输入号

HSC	使用的输入点	HSC	使用的输入点
HSC0	0 ~ 3	HSC3	6,7
HSC1	2,3	HSC4	8 ~ 11
HSC2	4 ~ 7	HSC5	10,11

由表 3-10 可见，HSC0 的副计数器是 HSC1，其他 HSC 类似。所以每组 HSC 都有共用的输入通道，表 3-11 列出 HSC 的输入使用本地 I/O 情况。

表 3-11　HSC 的输入使用本地 I/O 情况

HSC	本地 I/O											
	0	01	02	03	04	05	06	07	08	09	10	11
HSC0	A/C	B/D	Reset	Hold								
HSC1			A/C	B/D								
HSC2					A/C	B/D	Reset	Hold				
HSC3							A/C	B/D				
HSC4									A/C	B/D	Reset	Hold
HSC5											A/C	B/D

表 3-12 列出 Micro830/Micro850 的 24 点 HSC 的输入接线情况。

表 3-12　Micro830/Micro850 的 24 点 HSC 的输入接线情况

操作模式	Input0(HSC0) Input2(HSC1) Input4(HSC2) Input6(HSC3)	Input 1(HSC0) Input 3(HSC1) Input 5(HSC2) Input 7(HSC3)	Input2 (HSC0) Input6 (HSC2)	Input3 (HSC0) Input7 (HSC2)	用户程序中，模式的值
以内部方向计数（mode 1a）	增计数	未使用			0
以内部方向计数，外部提供 Reset 和 Hold 信号（mode 1b）	增计数	未使用	Reset	Hold	1
以外部方向计数（mode 2a）	增/减计数	方向	未使用		2
外部提供方向，Reset 和 Hold 信号（mode 2b）	计数	方向	Reset	Hold	3
两输入计数器（mode 3a）	增计数	减计数	未使用		4
两输入计数器，外部提供 Reset 和 Hold 信号（mode 3b）	增计数	减计数	Reset	Hold	5
正交计数器（mode 4a）	A 型输入	B 型输入	未使用		6
正交计数器，外部提供 Reset 和 Hold 信号（mode 4b）	A 型输入	B 型输入	Z 型 Reset	Hold	7
正交 X4 计数器（mode 5a）	A 型输入	B 型输入	未使用		8
正交 X4 计数器，外部提供 Reset 和 Hold 信号（mode 5a）	A 型输入	B 型输入	Z 型 Reset	Hold	9

其他点数不同的控制器 HSC 接线方式类似上表，主 HSC 可以使用 4 个输入端口，但是副 HSC 只能使用后两个输入端口，具体接线方式取决于操作模式。

3.3　Micro850 控制器扩展式模块

介绍了 Micro850 控制器的硬件特性和本地 I/O 设置以后，下面重点介绍 Micro850 控制器的扩展式模块，Micro850 控制器支持多种离散量和模拟量扩展 I/O 模块。可以连接任意组合的扩展 I/O 模块到 Micro850 控制器上，但要求本地、嵌入、扩展的离散量 I/O 点数小于或等于 132。Micro850 扩展模块见表 3-13。

表 3-13　Micro850 扩展模块

扩展模块型号	类别	种　　类
2085-IA8	离散	8 点，120V 交流输入
2085-IM8	离散	8 点，240V 交流输入
2085-OA8	离散	8 点，120/240V 交流晶闸管输出
2085-IQ16	离散	16 点，12/24V 拉出/灌入型输入
2085-IQ32T	离散	32 点，12/24V 拉出/灌入型输入
2085-OV16	离散	16 点，12/24V 直流灌入型晶体管输出

（续）

扩展模块型号	类别	种　类
2085-OB16	离散	16 点，12/24V 直流拉出型晶体管输出
2085-OW8	离散	8 点，交流/直流继电器型输出
2085-OW16	离散	16 点，交流/直流继电器型输出
2085-IF4	模拟	4 通道，14 位隔离电压/电流输入
2085-IF8	模拟	8 通道，14 位隔离电压/电流输入
2085-OF4	模拟	4 通道，12 位隔离电压/电流输出
2085-IRT4	模拟	4 通道，16 位隔离热电阻（RTD）和热电偶输入模块
2085-ECR	终端	2085 的总线终端电阻

模拟量扩展 I/O 模块是一种将模拟信号转化为数字信号输入到计算机以及将数字信号转化为模拟信号输出的模块。控制器可以通过这些控制信号达到控制的目的。

3.3.1　模拟量模块

1. I/O 属性

2085-IF4 和 2085-IF8 分别支持 4 通道和 8 通道输入，2085-OF4 支持 4 通道输出。每个通道都可以被设置成电流或者电压输入/输出，其中电流模式为默认配置。

2. 数据范围

2085-IF4、2085-IF8 和 2085-OF4 的有效数据范围见表 3-14、表 3-15。

表 3-14　2085-IF4、2085-IF8 的有效数据范围

数据类型	类型/范围			
	0～20mA	4～20mA	-10～10V	0～10V
原始/比例数据	-32768～32767			
工程单位	0～21000	3200～21000	-10500～10500	-500～10500
百分比范围	0～10500	-500～10625	不支持	-500～10500

表 3-15　2085-OF4 的有效数据范围

数据类型	类型/范围			
	0～20mA	4～20mA	-10～10V	0～10V
原始/比例数据	-32768～32767			
工程单位	0～21000	3200～21000	-10500～10500	0～10500
百分比范围	0～10500	-500～10625	不支持	0～10500

3. 数据配置

（1）2085-IF4 的数据配置

模拟量输入值能从全局变量_ IO _ Xx _ AI _ yy 读出，x 代表扩展槽 1～4，yy 代表通道数号 00～03。2085-IF8 和 2085-OF4 的模拟量输入值读取以及 x、y 意义同 2085-IF4。

模拟量输入状态值能从全局变量_ IO _ Xx _ ST _ yy 读出，x 代表扩展槽号 1 ~4，yy 代表状态字 00 ~02。2085-IF4 的数据格式见表 3-16。

表 3-16　2085-IF4 的数据格式

字	R/W	7	6	5	4	3	2	1	0
状态 0	R	保留							
状态 1	R	保留		HHA0	LLA0	HA0	LA0	DE0	S0
状态 2	R	保留		HHA2	LLA2	HA2	LA2	DE2	S2
字	R/W	15	14	13	12	11	10	9	8
状态 0	R	PU	GF	CRC	保留				
状态 1	R	保留	HHA1	LLA1	HA1	LA1	DE1	S1	
状态 2	R	保留	HHA3	LLA3	HA3	LA3	DE3	S3	

（2）2085-IF8 的数据配置

模拟量输入状态值能从全局变量 IO _ Xx _ ST _ yy 读出，x 代表扩展槽号 1 ~4，yy 代表状态字 00 ~04。若是想从状态字中读出某一位，也能通过在全局变量后面附加一个 zz 读出来，zz 代表位号 00 ~15。2085-IF8 的数据格式见表 3-17，2085-IF4 和 2085-IF8 的字段描述见表 3-18。

表 3-17　2085-IF8 的数据格式

字	R/W	7	6	5	4	3	2	1	0
状态 0	R	保留							
状态 1	R	保留		HHA0	LLA0	HA0	LA0	DE0	S0
状态 2	R	保留		HHA2	LLA2	HA2	LA2	DE2	S2
状态 3	R	保留		HHA4	LLA4	HA4	LA4	DE4	S4
状态 4	R	保留		HHA6	LLA6	HA4	LA4	DE4	S6
字	R/W	15	14	13	12	11	10	9	8
状态 0	R	PU	GF	CRC	保留				
状态 1	R	保留		HHA1	LLA1	HA1	LA1	DE1	S1
状态 2	R	保留		HHA3	LLA3	HA3	LA3	DE3	S3
状态 3	R	保留		HHA5	LLA5	HA5	LA5	DE5	S5
状态 4	R	保留		HHA7	LLA7	HA7	LA7	DE7	S7

表 3-18　2085-IF4 和 2085-IF8 的字段描述

字段	描　　述	
CRC	CRC 错误	当数据上的 CRC（循环冗余校验）错误发生时，此位置 1，当收到下一个正确数据时，此位清零
DE#	数据错误	当允许输入通道没有从当前采样中读到任何值时，此位置 1。各自的返回输入数据值仍和之前的值一样
GF	一般错误	当任何一种错误发生时，此位置 1

（续）

字段		描　　述
HA#	高位报警	当输入通道的值超过之前设置的高位报警值时，此位置 1
HHA#	高高位报警	当输入通道的值超过之前设置的高高位报警值时，此位置 1
LA#	低位报警	当输入通道的值低于之前设置的低位报警值时，此位置 1
LLA#	低低位报警	当输入通道的值低于之前设置的低低位报警值时，此位置 1
PU	上电	1，上电以后此位置 1，当模块接收到正确的设置以后，此位清零 2，当控制器在运行模式下发生意外重启的时候，此位置 1。同时通道故障位 S#也会置 1。模块在重启以后还未进行配置的状态下仍然是连接上的。当再次配置正确后，PU 和通道故障位 S#会清零
S#	通道故障	当相应的通道打开了，或者有错误，或者低于/高于界限时，此位置 1

（3）2085-OF4 数据配置

控制位状态值能从全局变量_ IO _ Xx _ CO _ 00zz 读出，x 代表扩展槽号 1 ~4，zz 代位号 00 ~12。2085-OF4 控制数据配置见表 3-19。

表 3-19　2085-OF4 控制数据配置

字	位　　号							
	7	6	5	4	3	2	1	0
控制 0	UU3	UO3	UU2	UO2	UU1	UO1	UU0	UO0
字	位　　号							
	15	14	13	12	11	10	9	8
控制 0	保留				CE3	CE2	CE1	CE0

UUx 和 UOx 在运行模式下置位是用来清除任何锁定的上下限警报。当解锁位置 1 并且报警的条件不存在时，警报被解除。如果仍然处在报警条件之下，那么解除报警位则无效。

CEx 在运行模式下置位可以清除任何 DAC 硬件错误位以及使得错误禁用的通道 x 再使能。

使用通道报警和错误解锁时，需要保持解锁位置位（1），直到正确的输入通道状态字告诉我们报警状态位复位了，此时再复位解锁位（0）。

（4）状态数据

模拟量输出状态值能从全局变量_ IO _ Xx _ ST _ yy 读出，x 代表扩展槽 1 ~4，yy 代表状态字 00 ~06。状态字中独立的每一位能够通过在全局变量的名字后面加 zz 读出来，zz 是位号 0 ~15。

2085-OF4 状态数据配置和字段描述见表 3-20、表 3-21。

表 3-20　2085-OF4 状态数据配置

字	位　　号							
	7	6	5	4	3	2	1	0
状态 0	通道 0 数据值							

（续）

字	位号							
	7	6	5	4	3	2	1	0
状态 1	通道 1 数据值							
状态 2	通道 2 数据值							
状态 3	通道 3 数据值							
状态 4	E3	E2	E1	E0	S3	S2	S1	S0
状态 5	保留		U1	O1	保留		U0	O0
状态 6	保留							
字	位号							
	15	14	13	12	11	10	9	8
状态 0	通道 0 数据值							
状态 1	通道 1 数据值							
状态 2	通道 2 数据值							
状态 3	通道 3 数据值							
状态 4	PU	GF	CRC	保留	保留			
状态 5	保留		U3	O3	保留		U2	O2
状态 6	保留							

表 3-21　2085-OF4 的字段描述

字段	描述	
CRC	CRC 错误	表明数据接收到错误，所有的通道错误位置位，当接收到下一个正确数据的时候，此位清零
Ex	错误	表明有一个硬件错误，包括坏线或者通道 x 的高负载，错误代码可能会在各自的输入字（0～3）上显示出来，相应的通道会被关闭，直到用户通过写输出数据的 CEx 位清除了错误，通道才会打开
GF	一般错误	表明发生错误了，包括 RAM 测试失败，ROM 测试失败，EEPROM 失败。此时，所有通道错误位 Sx 同样会置位
Ox	超范围标志	表明控制器正尝试驱动高于其正常操作范围或者通道的高钳位电平的模拟量输出。但是如果通道的高钳位电平等级没有设置的话，模块会持续将模拟量输出转化为最大量程的值
PU	上电	当控制器在运行模式下发生意外重启的时候，此位置 1。同时 Ex 和 Sx 也会置 1。模块在重启以后还未进行配置的状态下仍然是连接上的。当再次配置正确后，PU 和通道故障位会清零
Sx	通道错误	表明 x 通道上有一个错误
Ux	低于范围标志	表明控制器正尝试驱动低于其正常操作范围或者通道的低钳位电平（如果低钳位电平设置了的话）的模拟量输出

3.3.2　四通道 IRT4 模块

2085-IRT4 支持热电偶与热电阻的种类和电压、阻值范围见表 3-22、表 3-23。

表 3-22 热电偶的种类和电压范围

传感器类型	温 度 范 围	传感器类型	温 度 范 围
B	300 ~ 1800℃(572 ~ 3272 ℉)	N	-270 ~ 1300℃(-454 ~ 2372 ℉)
C	0 ~ 2315℃(32 ~ 4199 ℉)	R	-50 ~ 1768℃(-58 ~ 3214 ℉)
E	-270 ~ 1000℃(-454 ~ 1832 ℉)	S	-50 ~ 1768℃(-58 ~ 3214 ℉)
J	-210 ~ 1200℃(-346 ~ 2192 ℉)	T	-270 ~ 400℃(-454 ~ 752 ℉)
K	-270 ~ 1372℃(-454 ~ 2502 ℉)	mV	0 ~ 100mV
TXK/XK(L)	-200 ~ 800℃(-328 ~ 1472 ℉)		

表 3-23 热电阻的类型和阻值范围

传感器类型	温度范围	传感器类型	温度范围
100ΩPtα = 0.00385 Euro	-200 ~ 870℃(-328 ~ 1598 ℉)	200ΩNickel 618	-60 ~ 200℃(-76 ~ 392 ℉)
200ΩPtα = 0.00385 Euro	-200 ~ 400℃(-328 ~ 752 ℉)	120ΩNickel 672	-80 ~ 260℃(-112 ~ 500 ℉)
100ΩPtα = 0.003916 U. S	-200 ~ 630℃(-328 ~ 1166 ℉)	10ΩCopper(铜)427	-200 ~ 260℃(-328 ~ 500 ℉)
200ΩPtα = 0.003916 U. S	-200 ~ 400℃(-328 ~ 752 ℉)	Ohms(电阻)	0 ~ 500Ω
100ΩNickel(镍)618	-60 ~ 250℃(-76 ~ 482 ℉)		

2085-IRT4 模块使组态的每一个输入通道的模拟信号线性转换成温度值。通过 Connected Components Workbench 软件为通道 0 ~ 3 组态以下数据格式：

1）工程单位 *1：如果选择工程单位 *1 作为温度传感器的数据格式，对于确定的传感器类型和标准，模块将输入数据转换成实际温度值。表示的温度值是每单位 0.1℃/℉。对电阻式输入来说，每单位对应值为 0.1Ω。对电压式输入来说，每单位对应值为 0.01mV。

2）工程单位 *10：热电偶或电阻式输入传感器，对于确定的传感器类型和标准，模块将输入数据转换成实际温度值。对于这种格式，每单位表示的温度值是 1℃/℉。对电阻式输入来说，每单位对应阻值为 1Ω。对电压式输入来说，每单位对应电压值为 0.1mV。

3）成比例的数据模式：控制器输出的值与输入量成正比，数值的范围由 A/D 转换器的分辨率而定。例如：热电偶类型 B（300 ~ 1800℃）的数据范围是 -32768 ~ 32767。

4）百分比范围：在正常工作范围内，输入数据被描述成百分比。例如：热电偶 B 型传感器，0 ~ 100mV 等价于 0 ~ 100% 或 300 ~ 1800℃ 等价于 0 ~ 100%。

通道 0 ~ 3 的数据格式的有效范围见表 3-24。

表 3-24 通道 0 ~ 3 的数据格式的有效范围

数据格式	传感器类型-温度	传感器类型 0 ~ 100mV	传感器类型 0 ~ 500Ω
成比例的数据	-32768 ~ 32767		
工程单位 *1	温度值(℃/℉)	0 ~ 10000	0 ~ 5000
工程单位 *10	温度值(℃/℉)	0 ~ 1000	0 ~ 500
百分比范围	0 ~ 100%		

下面介绍如何将模拟量转换为特定数据格式对应的数值。

将模拟值 x 转换成计算数据 y 或将计算数据 y 转换成模拟值 x 的公式如下：

y =((x-x 数据范围最小值)*(y 数据范围)/(x 数据范围))+y 数据范围最小值

例如：

x = -2000（未处理的数据），x 最小值 = -32768，x 数值范围 = 32767 -(-32767)= 65535，

y 数值范围 = 1372 -(-270)= 1642，y 最小值 -270℃，

则：y =(-20000 -(-32768))*1642/65535 +(-270℃)= 49.9℃

温度单位可以设为℃（默认）或℉。

接下来介绍其他概念：

开路响应：这个参数定义了模块处于开路时的响应。

数值上限：通道数值的输入最大值。数值上限由选择的输入类型、数据格式和数据范围决定。

数值下限：通道数值的输入最小值。数值下限由选择的输入类型、数据格式和数据范围决定。

保持上一状态：将输入设为上一输入值。

调零：调输入为零，强制通道数据为零。

滤波器频率：

2085-IRT4 模块用一个数字滤波器为输入信号抑制干扰。频率默认值为 4Hz。当滤波频率为 4Hz 时，数字滤波器提供 -3dB 的振幅衰减。

-3dB 频率是滤波器的截止频率（截止频率定义：输入信号的频率响应曲线 3dB 点对应的频率即截止频率）。所有小于截止频率的输入信号都会以小于 3dB 衰减量通过数字滤波器。所有高于截止频率的输入信号都会加速衰减。

每一个通道的截止频率由滤波器频率的选择来确定，并且等于滤波器的设定频率。选择一个滤波器之后，信号的最快变化速率低于滤波器的截止频率。截止频率不能和响应时间混淆。截止频率和输入信号的数字滤波器的衰减程度有关。响应时间定义为：扫描完输入通道后通道数据字的更新速度。

补充说明：低通滤波器的抑制噪声效果更好，但是响应时间通常会增加。高通滤波器的响应速度很快，但会减弱抑制噪声的效果，并且降低有效分辨率。

3.4　添加扩展式 I/O

当在 CCW 中运用“检测”功能时，扩展的 I/O 模块就会自动添加到项目中。在已建立的 Micro850 控制器中添加一个 I/O 扩展式模块，可以按照以下步骤做：

1）在工程管理器窗口中，右击 Micro850 并且单击 Open，如图 3-6 所示。

在原来的 Mirco850 控制器的主窗口中会在第一级出现图片代替 Mirco 控制器，第二级是控制器属性，最后一级是输出框。CCW 中 Micro850 界面如图 3-7 所示。

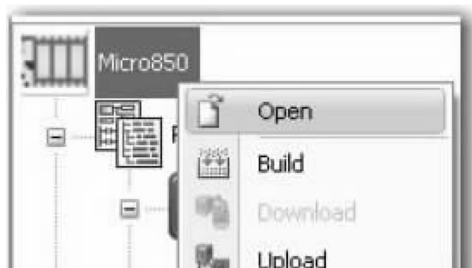


图 3-6　打开 Micro850 控制器

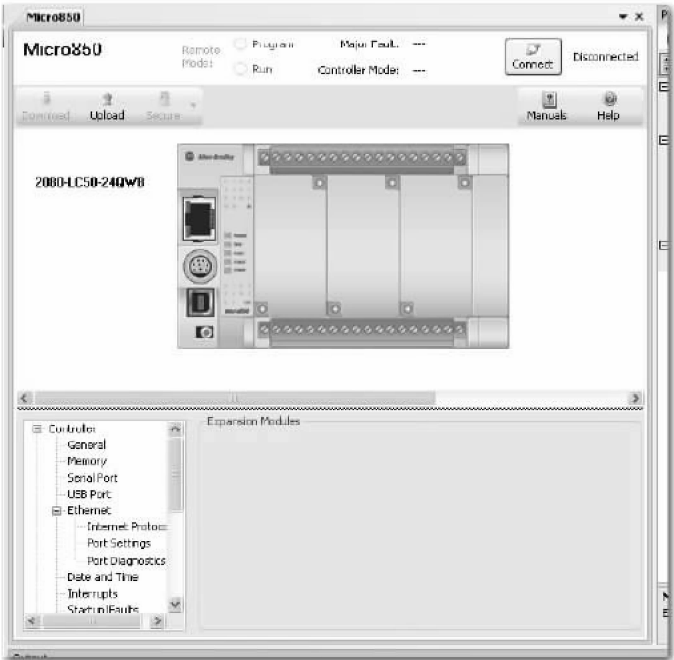


图 3-7　CCW 软件中 Micro850 界面

2）在 CCW 软件窗口最右边的角落，也就是设备工具箱窗口可以找到扩展模块文件夹，模块选择如图 3-8 所示。

3）点击并拖动把 2085-IQ32T 放在控制器右边第一个槽口中。4 个蓝色的槽表示可以用来安放 I/O 扩展模块的槽口。

4）在设备工具箱的窗口的扩展模块文件夹里，拖动并把 2085-IF4 放在第二个 I/O 扩展槽中，就是 2085-IQ32T 的旁边。

5）拖动并把 2085-OB16 放在第三个 I/O 扩展槽。

6）拖动并把 2085-IRT4 放在第四个 I/O 扩展槽。

在添加了 4 个 I/O 扩展模块之后，CCW 工程应该如图 3-9 所示。

注意，2085-IQ32T 模块本身不带接线口，所以这里提供有 3 种接线方式：一种是用 40 针连接线连接到端子排接线；第二种是使用 Allen-Bradley 1492 连接电缆引线；第三种是使用 Allen-Bradley 1492 连接电缆连接到端子排上。

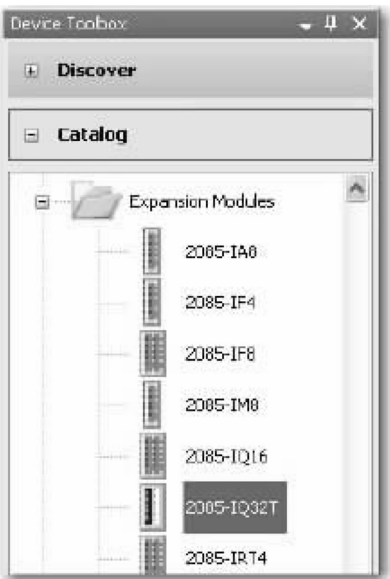


图 3-8　模块选择

扩展模块组态属性如图 3-10 所示。点击 General，可以看到刚刚添加的每个扩展 I/O 设备的详情。点击 Configuration（如果需要），观察默认组态属性。

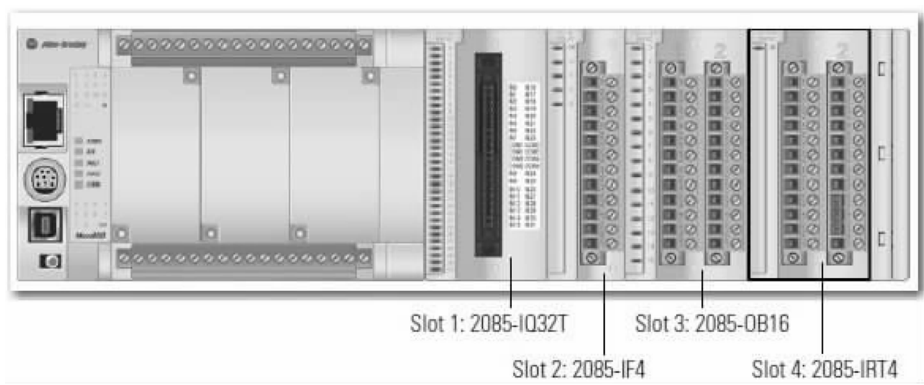


图 3-9　添加 4 个 I/O 扩展模块

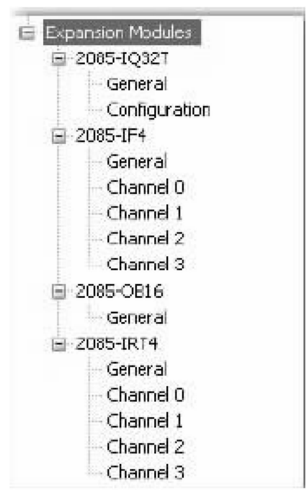


图 3-10　扩展模块组态属性

3.5　编辑扩展 I/O 组态

在控制器图像下方可以通过扩展模块的详情框来编辑默认 I/O 组态。

1）选择想要组态的 I/O 扩展设备。扩展模块详情如图 3-11 所示。

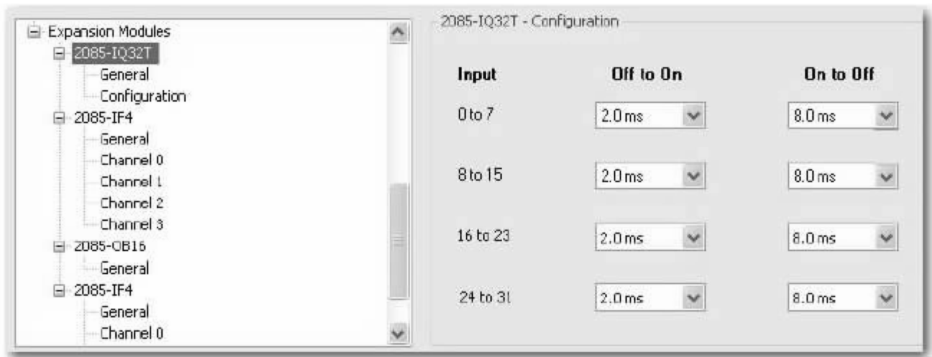


图 3-11　扩展模块详情

2）点击 Configuration。根据要求和应用来编辑模块和通道的属性。下一部分为扩展 I/O 模块组态属性。

在 CCW 软件中交流输入模块 2085-IA8 和 2085-IM8 对用户来说只有普通设备的功能，没有组态属性。设备功能如图 3-12、图 3-13、图 3-14 所示。

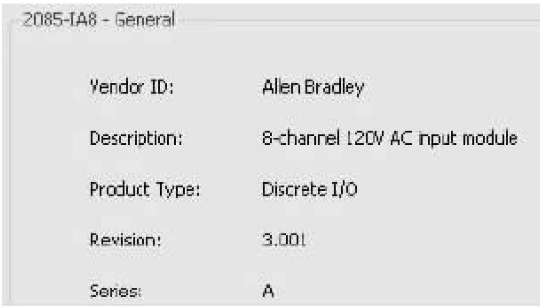


图 3-12　2085-IA8 设备功能

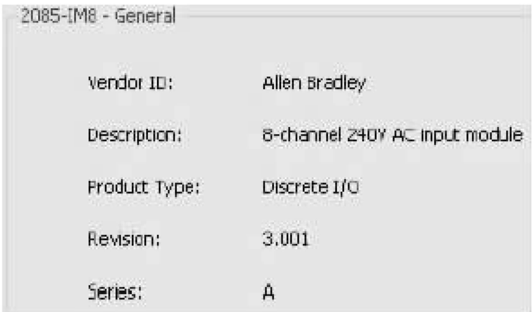


图 3-13　2085-IM8 设备功能

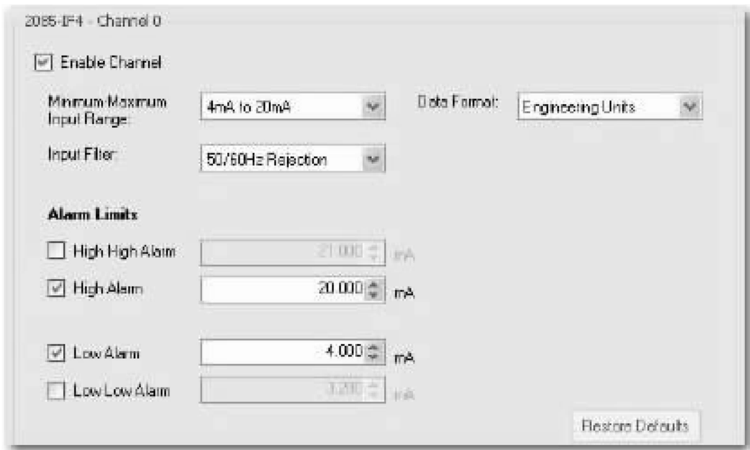


图 3-14　2085-IF4 组态属性

对于 2085-IF4 和 2085-IF8 这两个模拟输入模块，可以对以下属性进行设定，例如：输入范围、数据格式、滤波以及每个通道的报警限值。

2085-IF4 和 2085-IF8 的组态参数见表 3-25。

表 3-25　2085-IF4 和 2085-IF8 的组态参数

组态属性	操　作　内　容	描　　述
使能通道	选择或不选择复选框。框格是默认选择的	通过这个复选框使能或不使能一个通道，默认的，每个通道是使能的
最小-最大输入限度	值的范围选择如下： ● 0～20mA ● 4～20mA（默认） ● －10～10V ● 0～10V	设置每个通道的输入模式，不是电压就是电流，电流一般设为默认模式

（续）

组态属性	操　作　内　容	描　　述
数据格式	从下面的选项中选择： ● 原始/比例数据 ● 工程单位（默认） ● 百分比范围	
输入滤波	在下面的选项中选择： 50/60Hz Rejection No Filter 2-Point Moving Average 4-Point Moving Average 8-Point Moving Average 50/60Hz Rejection	
高高报警 高报警 低报警 低低报警	检查复选框以便使能一个报警。默认的，高高和低低报警是不工作的，高和低报警是工作的	当模块超过了每个通道所组态的高、高高、低、低低极限值时，过程状态报警就会启动

IQ32T 组态属性如图 3-15 所示。

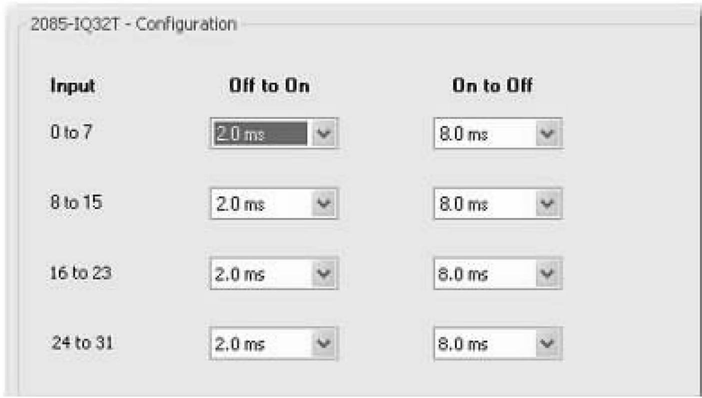


图 3-15　IQ32T 组态属性

这两个模块分别是 16 和 32 通道的直流输入模块，可以分别设定 OFFtoON 和 ONtoOFF 的范围。模块选择时间见表 3-26。

表 3-26　模块时间选择

输　　入	具　体　操　作	输　　入	具　体　操　作
OFF to ON	在下面的选项中选择： 8.0ms；4.0ms；2.0ms（默认） 1.0ms；0.5ms；0.1ms；0.0ms	ON to OFF	在下面的选项中选择： 8.0ms（默认）；4.0ms；2.0ms 1.0ms；0.5ms；0.1ms；0.0ms

2085-OV16、2085-OB16、2085-OW16、2085-OA8、2085-OW8 这 5 个模块均是输出模块，并且用户能在 CCW 里获得它们具体的设备信息。在 CCW 软件里没有这些模块的用户配置页面。2085-OF4 和 2085-IRT4 的配置页面如图 3-16、图 3-17 所示。



图 3-16　2085-OF4 配置页面

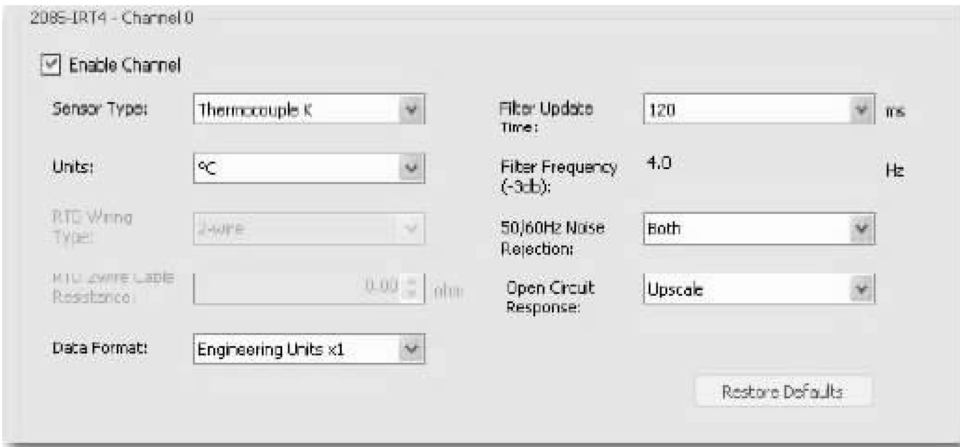


图 3-17　2085-IRT4 配置页面

2085-OF4 是模拟量输出模块，可以设定它的输出单元、最小到最大的输出范围、高钳位和低钳位的值以及超量程和欠量程值。2085-OF4 的组态参数见表 3-27。

表 3-27　2085-OF4 的组态参数

组态属性	具体操作	描　述
使能通道	选择或取消选择复选框。通道默认的状态是非使能	通过复选框来使一个通道工作或不工作。默认情况下，每个通道是不工作的
最小-最大输出范围	从下面的选项中选择： ● 0～20mA；● 4～20mA（默认） ● －10～10V；● 0～10V	

（续）

组态属性	具体操作	描　　述
数据格式	从下面的选项中选择： ●原始/比例数据；●工程单位（默认） ●百分比范围	
高钳位值	点击复选框以便键入一个高钳位值	一旦模块的高钳位和低钳位被设定，任何来自控制器的数据，如果超出了钳位设定，就会启动相应的警报，并且将输出值转化为相应钳位值
低钳位值	点击复选框以便键入一个低钳位值	
超限值警报触发器	如果开通并键入一个高钳位的值，可以通过 overrange alarm trigger. 来检查 High Clamp Value 如果未开通并键入一个高钳位的值，可以通过 overrange alarm trigger. 来检查 Maximum Output Value	当输出模块输出超过设定的输出极限时，超限和欠限的特征就能检测到。根据设定的钳位值或最小/最大输出值，相应的触发器会被触发
欠限警报触发器	如果开通并键入一个低钳位的值，可以通过 Underrange alarm trigger. 来检查并设定 Low Clamp Value 如果没有开通并键入一个低钳位的值，可以通过 Underrange alarm trigger. 来检查并设定 Minimum Clamp Value	
超限/欠限锁定	点击锁定	在设置位置上检查框格以便锁住警报，尽管导致报警的情况可能消失
恢复默认值	点击恢复默认值的按钮	恢复默认设备的属性

对于 RTD 和热电偶扩展 I/O 2085-IRT4，可以为任一通道设定传感器类型、数据格式、温度单元和其他属性。2085-IRT4 的组态参数见表 3-28。

表 3-28　2085-IRT4 的组态参数

组态属性	具 体 操 作	描　　述
使能通道	点击使能按钮	这个参数可以使特定的通道运行
传感器类型	从下面的选项中选择： ● 100 Platinum（铂）385；● 200 Platinum 385 ● 100 Platinum 3916；● 200 Platinum 3916 ● 100 Nickel（镍）618；● 200 Nickel 618 ● 120 Nickel 672；● 100 Copper（铜）427 ● 0～500Ω；● 0～100 mV ●热电偶 B；●热电偶 C；●热电偶 E ●热电偶 J；●热电偶 K； ●热电偶 TXK/XK（L）；●热电偶 N ●热电偶 R；●热电偶 S；●热电偶 T	为每个通道设定 RTD 或传感器的类型
单位	设定℃和℉	为每个通道设定温度的单位

（续）

组态属性	具 体 操 作	描 述
热电阻的配线类型	设定如下的类型： ● 2 线制；● 3 线制；● 4 线制	通道 x 的配线类型。这种数据只有当通道的传感器类型是热电阻或 {0 ~ 500Ω}
RTD 两线制电缆电阻	代替值从 0.0 ~ 500.00Ω 到 0.0 ~ 655.35Ω	两线制电缆规定了电缆电阻。当 RTD 的两线制电缆电阻值小于输入值时，输入电阻值会在每次读取时相应地减去多出的部分。当 RTD 的两线制电缆电阻值大于输入值时，欠量程或开放位被置（1）。设定电缆阻值时，传感器的类型必须是 RTD 或 {0 ~ 500Ω}，并且 RTD 的接线类型必须是两线制。否则这个参数就是不可用的
数据格式	从下面的选项中选择： 原始/比例数据；工程单位 * 1 工程单位 * 10；百分比范围	
滤波器的更新时间/ms	按照下面的设置： 4；8；16；32；40；48；60；101； 120；160；200；240；320；480	注意：4ms 的滤波器更新时间不能用在 B，R，S，E，J，C，K，L，N 型的热电偶传感器，或者 0 ~ 10mV 的传感器上 8ms 的滤波器更新时间不能用在 B，R，S 型的热电偶传感器上
过滤更新频率（ -3dB）	按照下面设置（用 Hz 为单位）： 114；60；30；14；12；9.4；8.0 4.7；4.0；3.0；2.4；2.0；1.5；1.0	
50/60Hz 噪声抑制	如下设定： ● 都选用（默认）；● 只选 50 ● 只选 60；● 都不选	
开放式电路响应	从下面的选项中选择： ● 高档 ● 低档 ● 保持最后状态 ● 零	下面定义开放式电路的响应类型： Upscale（高档）-将输入设置成通道的满量程，满量程的值由输入类型、数据格式和标度决定 Downscale（低档）-将输入设置成通道的量程下限，量程下限的值由输入类型、数据格式和标度决定 Hold Last State-设定输入来持续输入值 Zero-设定输入值为 0 来使每个通道的数据值为 0

3.6 删除和更换扩展 I/O 组态

通过前节实例，可以删除槽 2 和槽 3 的 2085-IF4 和 2085-OB16。然后，相应地用 2085-OW16 和 2085-IQ32T 模块来分别替换到槽口 2、3。操作如下：

1）在主窗口的控制器图像上，右击模块 2085-IF4 并且点击 Delete，删除模块示意图如图 3-18 所示。

2）出现一个是否清除模块占位符的对话框，如图 3-19 所示，点击“No”。

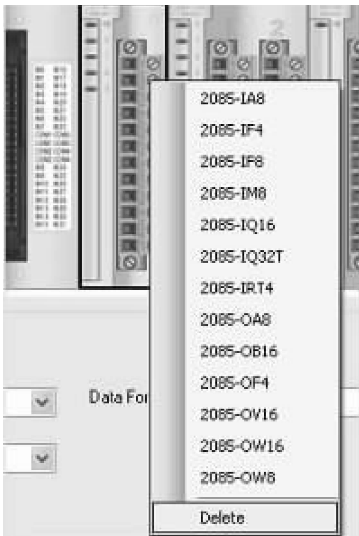


图 3-18　删除模块示意图

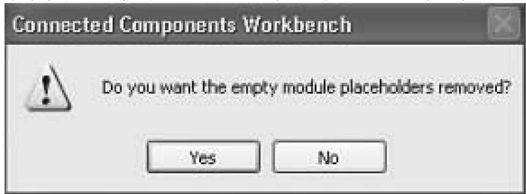


图 3-19　对话框

当从槽口 2 中删除 2085-IF4 时，图像会像图 3-20 一样。

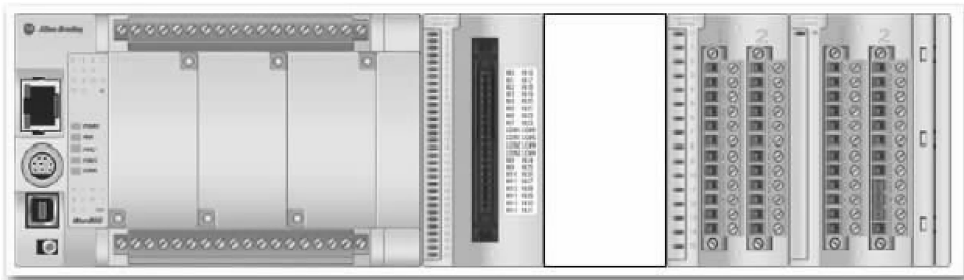


图 3-20　删除模块后的界面图

3）在空槽中（槽 2），右击并选择 2085-OW16。

4）下一步用 2085-IQ32T 设备代替槽 3 中的 2085-OB16。在槽 3 中右击 2085-OB16，并且选择 2085-IQ32T，如图 3-21 所示。

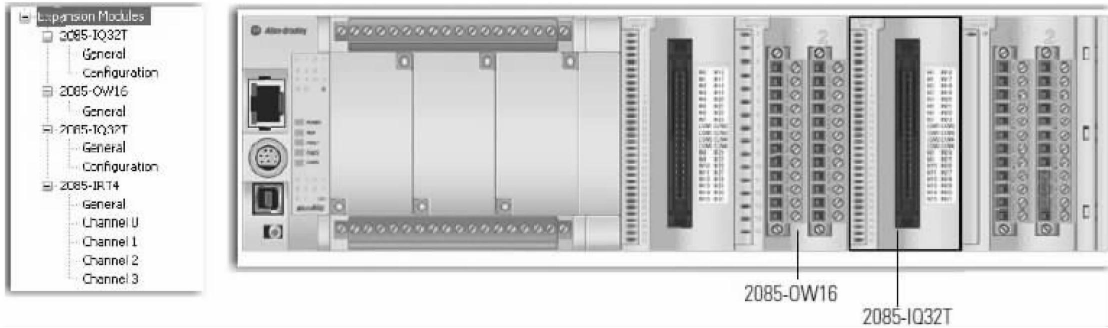


图 3-21　替换模块示意图

提示：也可以通过扩展模块列表来删除和更替扩展 I/O 模块。相应地，右击需要更换的 I/O 扩展模块，然后选择列表中你想要选用的 I/O 扩展模块来代替之前删除的。如果要删除就选择 Delete。

3.7 小结

本章描述了 Micro850 控制器的特点，并重点介绍了控制器的硬件特性。通过对 Micro850 控制器的组态，掌握和学习了对 Micro850 控制器的使用。由于这种微型的可编程序控制器推向市场及应用时间并不是很长，特别是它还在不断的升级和更新换代，很多功能大家还不是非常熟悉，因此十分需要对它的认识和学习。Micro850 控制器的综合能力极强，它不仅将 I/O 点设在处理器上，并且有高速计数模块、高速脉冲输出模块和两轴运动控制模块可供扩展。并且该控制器编程采用 USB 电缆，方便编程人员的使用，控制器除了本身具有串行通信端口，支持 MODBUS 和 ASCII 码通信协议以外，还有串行通信模块可供扩展，这使得网络通信功能更具有灵活性。与 Micro830 相比，通信上它可以通过以太网进行通信，方便快速。可以说 Micro800 控制器的功能正变得越来越强大，应用领域也会更加广泛。

习　题

1. Micro850 控制器有哪些硬件特性？
2. Micro850 控制器支持的电源类型有哪些？
3. Micro850 控制器的输入/输出有几种类型？
4. Micro850 控制器的本地 I/O 有哪些功能？
5. Micro850 控制器有几种通信端口？
6. Micro850 控制器可嵌入什么类型的模块？最多能嵌入几个？
7. Micro850 控制器可外挂什么类型的模块？最多能外挂几个？
8. Micro850 控制器支持哪些嵌入式、扩展式模块？
9. Micro850 控制器的 I/O 配置是怎样的？

思考题

1. 请用 Micro850 控制器设计一个简单的运动控制系统。
2. 比较 Micro850 和 Micro830、Micro810 控制器，说说各自的优缺点。

第 4 章

Micro800 控制器的网络通信

学习目标

- 了解罗克韦尔自动化的 NetLinx 网络体系结构
- 掌握使用 USB 与电脑通信的方法
- 掌握 Micro 800 控制器的 RS-485 通信
- 掌握使用 RS-232 串口通信的方法
- 掌握使用 Ethernet 通信的方法

4.1 Micro800 控制器的网络结构

罗克韦尔自动化公司的新一代微型 PLC Micro800 系列，随同一体化编程和组态软件 Connected Components Workbench（CCW），制定了一整套新的 PLC 和软件标准。Micro800 系列 PLC 主要用于经济型单机控制，结构和功能相对简单，因此其网络结构也不复杂。

Micro800 系列 PLC 支持 RS-232 和 RS-485 基本串口，RS-232 和 RS-485 是电子工业协会（EIA）的标准，指定电气和机械串行二进制通信的特点。通过这些串口，Micro800 可以与一些通信设备进行串行通信。此外，Micro800 系列 PLC 自带用于编程的 USB 接口，可用于与计算机通信。

Micro830 及 850 系列 PLC 以及嵌入式的通信模块支持的串行端口协议有 Modbus RTU Master and Slave，ASCII（仅 RS-232）。Modbus 是一种半双工，主从通信协议。后续更新的控制器及软件还支持 CIP Serial Server（仅 RS-232）（Common Industrial Protocol，CIP）。Modbus 协议允许主设备最多与 247 个从设备进行通信。ASCII 可以使 Micro800 控制器连接到其他 ASCII 设备，如条形码阅读器、磅秤、串行打印机或其他智能设备。

Micro850 系列 PLC 支持基本 EtherNet/IP 端口，可以接入 EtherNet 中，与其他通信设备进行通信。该通信口支持的以太网协议有：EtherNet/IP Server、Modbus/TCP Server、DHCP Client。

此外，Micro830 和 Micro850 还支持以下协议：Modbus/TCP Server、CIP Symbolic Server。Micro800 控制器的网络结构如图 4-1 所示。

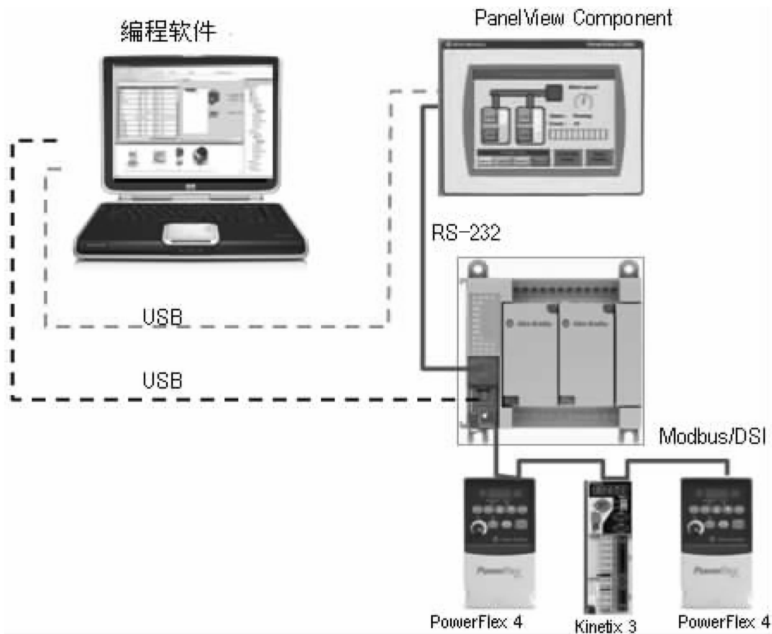


图 4-1　Micro800 控制器的网络结构

4.2　USB 通信

把 USB 电缆分别连接到控制器和计算机的 USB 接口上，当控制器和计算机第一次连接时，连接后会自动弹出安装 USB 连接驱动窗口，选择第一个选项，点击“下一步”。USB 安装成功后，在开始菜单中的所有程序找到 Rockwell Automation/CCW/Connected Components Workbench，点击进入软件。

在 Catalog 中选择所使用的 PLC 型号(2080-LC30-24QWB)，然后双击控制器图标。点击右上方的“Connect”按钮，弹出如图 4-2 所示的连接对话框，找到要连接的控制器型号，这里选择 Micro830 控制器，点击“OK”即可连接事先准备好的 2080-LC30-24QWB 控制器。

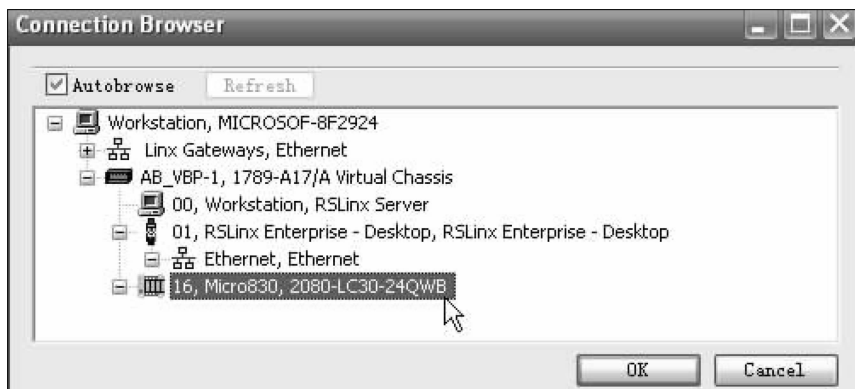


图 4-2　选择要连接的控制器

控制器连接计算机以后，“Disconnected”按钮为活动状态，同时可以改变控制器的状态。这样就完成了 Micro830 通过 USB 与计算机的通信，此时，可通过计算机对控制器下载程序。

4.3　RS-485 通信

由于 Micro830 控制器本身不带有 RS-485 接口，所以要用到 1761-NET-AIC 接口转换模块或 2080-SERIALISOL 模块来实现 RS-485 接口的连接。

下面的实验应用 2080-SERIALISOL 模块实现 Micro830 与 PowerFlex 4M 变频器的连接。系统硬件连接如图 4-3 所示。

有关 RS-232/485 隔离串口模块 2080-SERIALISOL，请参阅本书 2.3.6 节。RJ45 接头与正常的以太网线线序对应情况为：蓝线对应 4 号接头，蓝白线对应 5 号接头。4 号接头连接 485 正端（+485），5 号接头连接 485 负端（-485）。

连接完成后，需要设置 Micro830 控制器以及变频器参数，然后对控制器进行编程，输出控制命令给变频器，用来控制电动机。

首先，设置控制器参数，按照上一节讲述的方法，使 Mirco830 通过 USB 与计算机通信。然后用鼠标右键单击添加模块处，选择 2080-SERIALISOL 模块，如图 4-4 所示。

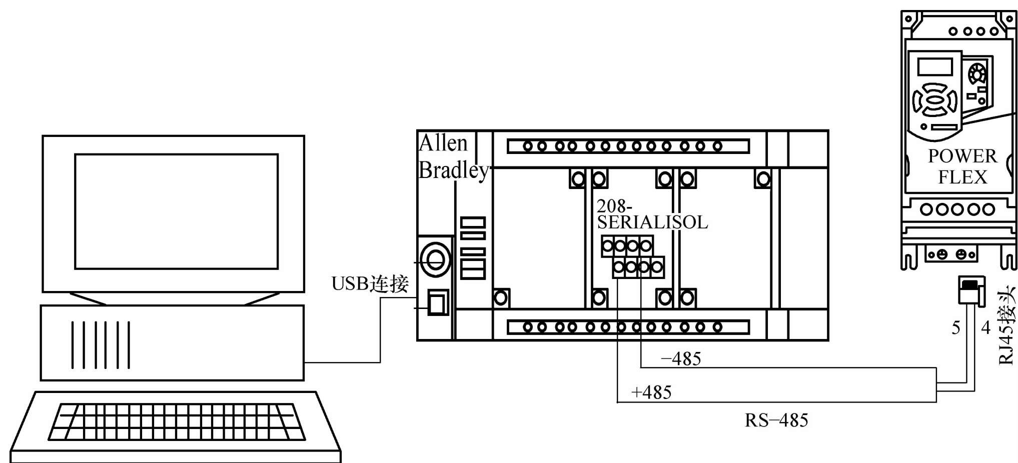


图 4-3　系统硬件连接图

添加完成后，在屏幕右下方的 Properties 选项处，设置模块的相应信息。由于控制器与变频器连接应用 Modbus 协议，所以选择 Modbus RTU 驱动，而控制器为主站，在最下面的一个选项选择 Modbus RTU Master。波特率选择 19200，无奇偶校验，单位地址设置为 1，如图 4-5、图 4-6 所示。

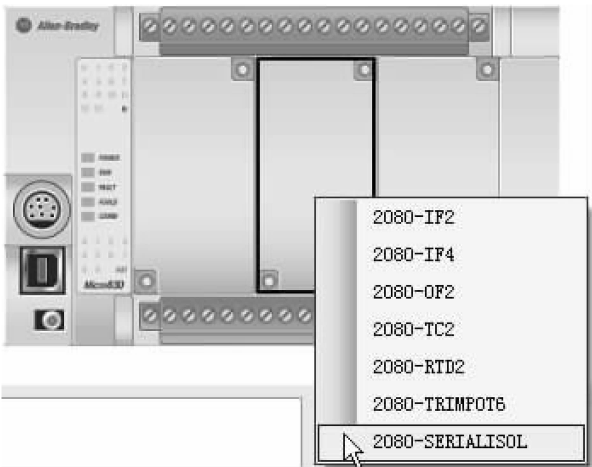


图 4-4　添加 2080-SERIALISOL 模块

驱动程序:	Modbus RTU
波特率:	19200
奇偶校验:	无
单位地址:	1
Modbus 角色:	Modbus RTU 主站

图 4-5　选择相关参数

高级设置

协议控制

媒介:	RS485	RTS 预延迟:	0
数据位:	8	RTS 后延迟:	0
停止位:	1		
响应计时器:	1000		
广播暂停:	1000		
字符间超时:	0		

图 4-6　高级选项设置

注意：设置完 2080-SERIALISOL 模块信息后即可对 Micro830 控制器进行编程，无需组态变频器，设置变频器参数通过控制面板进行，操作参见第 7 章 7.2 节。

通过变频器操作面板设置变频器参数，需要设置的参数见表 4-1。

表 4-1　变频器相关参数设置

参　　数	参数名称	设　　置	参　　数	参数名称	设　　置
P036	启动源	5 = 通信端口	A103	通信数据速率	4 = 19.2kbit/s
P038	速度参考频率	5 = 通信端口	A104	通信节点地址	100
A051	数字量输入 1 选项	6 = 通信端口	A107	通信格式	0 = RTU 8-N-1
A052	数字量输入 2 选项	0 = 不使用			

设置完成后，对控制器进行编程，本实验编写的程序使用了 3 个 MSG _ MODBUS 指令（请参考第 6 章 MSG _ MODBUS 指令），如图 4-7 所示。

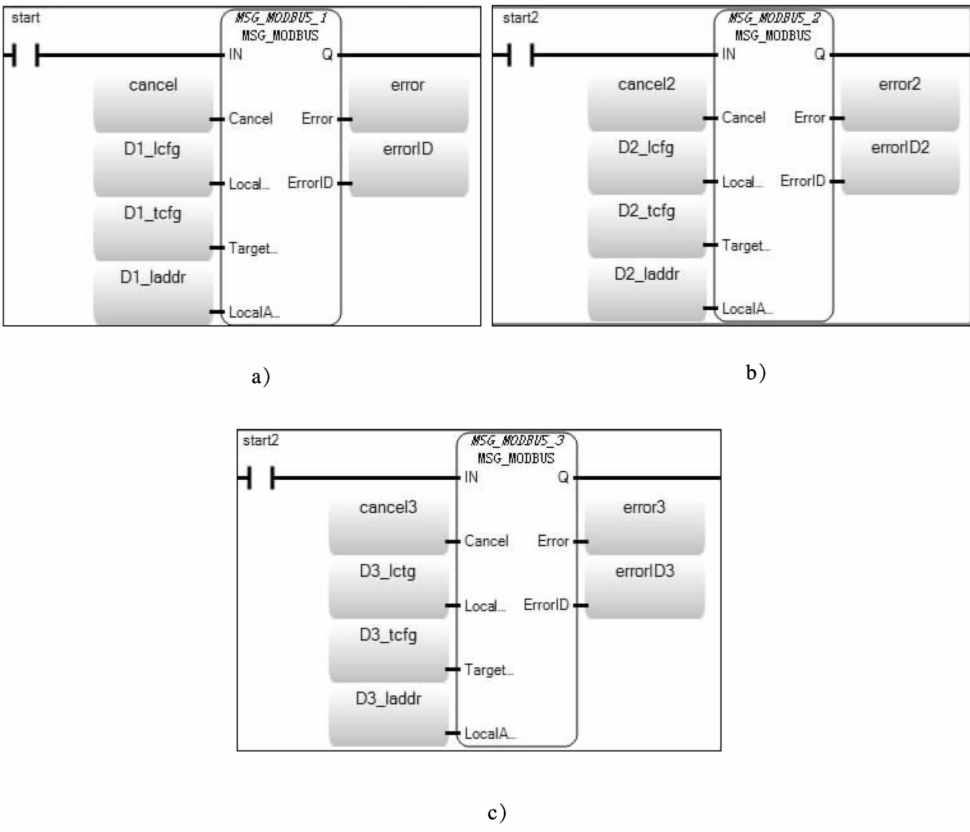


图 4-7　MSG _ MODBUS 程序

a）读取变频器反馈信息　b）启动电动机转动　c）设置变频器频率输入值为 50

然后下载并编译程序，打开程序数据列表，如图 4-8 所示。

D1_lcfg	...
D1_lcfg.Channel	5
D1_lcfg.TriggerType	0
D1_lcfg.Cmd	3
D1_lcfg.ElementCnt	4
D1_tcfg	...
D1_tcfg.Addr	8449
D1_tcfg.Node	100
D1_laddr	...
D1_laddr[1]	0

a)

D2_lcfg	...
D2_lcfg.Channel	5
D2_lcfg.TriggerType	0
D2_lcfg.Cmd	6
D2_lcfg.ElementCnt	1
D2_tcfg	...
D2_tcfg.Addr	8193
D2_tcfg.Node	100
D2_laddr	...
D2_laddr[1]	18

b)

D3_lctg	...
D3_lctg.Channel	5
D3_lctg.TriggerType	0
D3_lctg.Cmd	6
D3_lctg.ElementCnt	1
D3_tcfg	...
D3_tcfg.Addr	8194
D3_tcfg.Node	100
D3_laddr	...
D3_laddr[1]	500

c)

图 4-8　程序数据列表

a）读取变频器反馈信息程序相应参数　b）启动电动机转动程序相应参数

c）设置变频器频率输入值程序相应参数

这里，仅对读取变频器反馈信息程序进行讲解：

D1 _ lcfg 为本地设备定义结构输入，其包含四项：

1）D1 _ lcfg _ Channel 为通道地址，本实验即为 SERIALISOL 模块地址。

2）D1 _ lcfg _ TriggerType 为触发方式，0 为由假到真触发。

3）D1 _ lcfg _ Cmd 设置为 3，为读寄存器指令。

4）D1 _ lcfg _ ElementCnt 设置为 4 个字，为读取数据长度。

D1 _ tcfg 为目标设备定义结构输入，包含两项：

1）D1 _ tcfg _ Addr 为目标数据起始地址，8449 为变频器逻辑状态信息的 Modbus 寄存器地址 +1，具体可参照表 4-2。

表 4-2　Modbus 寄存器地址定义

寄存器地址（十进制）	相应位	说　　明
8192（逻辑命令字）	0	1 = 停止，0 = 不停止
	1	1 = 起动，0 = 不起动
	2	1 = 慢进，0 = 不慢进
	3	1 = 清除错误，0 = 不清除错误
	5，4	00 = 无命令设置；01 = 正转设置；02 = 反转设置； 11 = 无命令设置
	6，7	未使用

（续）

寄存器地址（十进制）	相应位	说明
8192（逻辑命令字）	9，8	00 = 无命令设置；01 = 使能加速度 1； 10 = 使能加速度 2 ；11 = 保持所选加速度
	11，10	00 = 无命令设置；01 = 使能减速度 1； 02 = 使能减速度 2；11 = 保持所选减速方式
	14，13，12	000 = 无命令设置
		001 = 频率源 = P038［Speed Source］
		010 = 频率源 = A069［Internal Freq］
		011 = 频率源 = 通信（地址 8193）
		100 = A070［Preset Freq 0］
		101 = A071［Preset Freq 1］
		110 = A071［Preset Freq 2］
		111 = A073［Preset Freq 3］
	15	1 = MOP 减少，0 = 不减少
8193（速度给定值）	十进制频率输入值（注意：对于 PowerFlex4、4M&40，十进制数中包含 1 个已固定的小数点。例如输入十进制数 100 表示设置频率为 10. 0Hz；对于 PowerFlex400，十进制数中包含两个已固定的小数点，例如输入十进制数 100 表示设置频率为 1. 0Hz）	
8448（逻辑状态字）	0	1 = 准备好，0 = 未准备好
	1	1 = 运行状态，0 = 没有运行
	2	1 = 正转命令，0 = 反转命令
	3	1 = 正转状态，0 = 反转状态
	4	1 = 加速状态，0 = 非加速状态
	5	1 = 减速状态，0 = 非减速状态
	6	1 = 警告，0 = 无警告
	7	1 = 故障状态，0 = 非故障状态
	8	1 = 达到速度参考值，0 = 未达到速度参考值
	9	1 = 速度参考值由通信端口控制
	10	1 = 操作命令由通信端口控制
	11	1 = 参数处于锁定状态
	12	数字输入 1 状态
	13	数字输入 2 状态
	14，15	未使用
8451（速度反馈值）	十进制频率反馈值（注意：对于 PowerFlex4、4M&40，十进制数中包含 1 个已固定的小数点。例如输入十进制数 100 表示设置频率为 10. 0Hz；对于 PowerFlex400，十进制数中包含两个已固定的小数点，例如输入十进制数 100 表示设置频率为 1. 0Hz）	

2）D1 _ tcfg _ Node 为目标设备节点地址，之前设置变频器参数时节点地址设为 100，这里的值也相应设为 100。

D1 _ laddr 为接收的数据，即读取的反馈信息放到这里。

注意，寄存器地址偏移量为 1，例如：逻辑命令的寄存器地址是 8192，而实际操作中就要设置为 8193。

将 start 位置 1，发现 D1 _ laddr 出现了参数 0，即从寄存器中读取的状态反馈信息，再将 start2 位置 1，发现电动机起动，改变 D3 _ laddr（1）中的数据，即能对电动机转速进行控制。

至此，就完成了 Micro830 通过 RS-485 组态变频器的实验。

4.4 RS-232 串口通信

Micro830 控制器本身自带 RS-232 串口，可以和支持 Modbus 协议的一些设备通过 RS-232 通信，本节通过一个实验来研究 Micro830 的 RS-232 串口通信。

本实验中，Micro830 通过 RS-232 与 Panel View 600 通信。硬件接线图如图 4-9 所示。

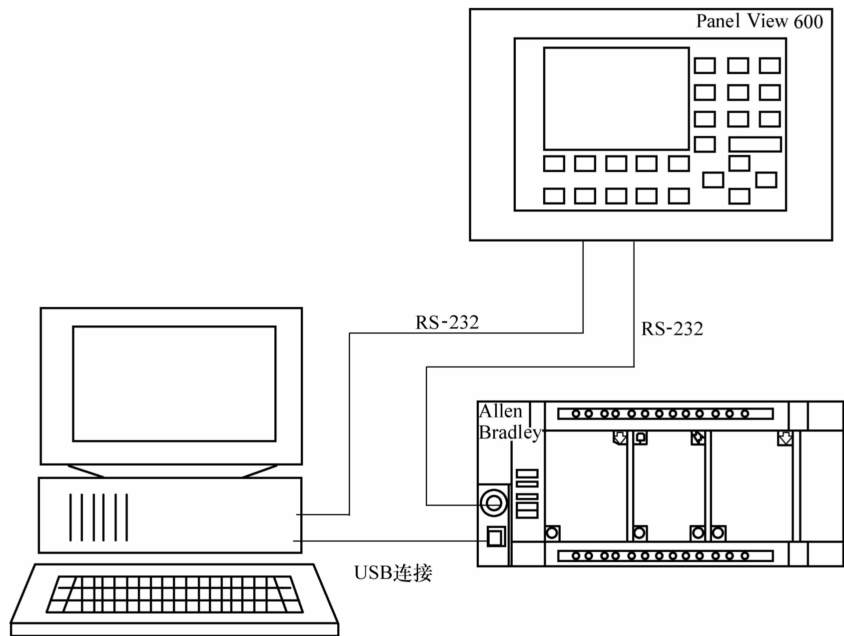


图 4-9 RS-232 实验的硬件接线图

PannelView 与 Micro830 通过用于通信的 RS-232 串口进行连接，PannelView 通过专门用于打印和程序下载的 RS-232 串口连接计算机，控制器通过 USB 与计算机连接，用于程序下载。

硬件连接完成后，打开 Connected Components Workbench 软件，按照前面所讲的方法，选择 2080-LC30-24QWB 控制器型号，打开控制器界面，在屏幕左下方找到 Communication Ports/Serial Port。单击出现如图 4-10 ~ 图 4-12 所示界面，在这里对串口的通信进行配置。

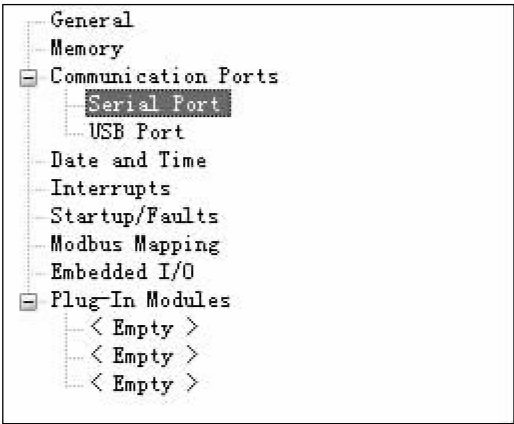


图 4-10　选择 Serial Ports

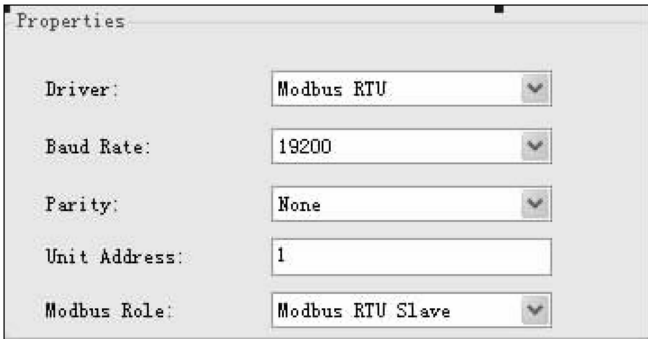


图 4-11　Micro830 通信配置界面

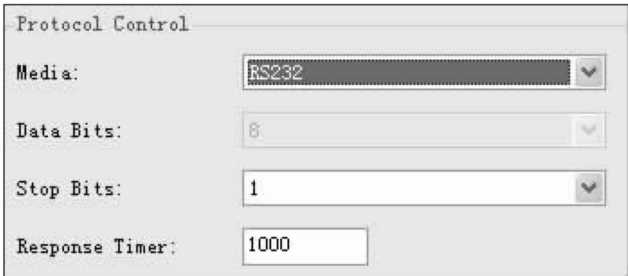


图 4-12　高级设置界面

由于应用的是 Modbus 协议，所以 Driver 一项选择 Modbus RTU，Baud Rate 选择 19200，Parity 选择 None，节点地址选择 1。由于本实验控制器在 Modbus 中处于从属地位，所以 Modbus Role 一项选择 Modbus RTU Slave，在下面的高级选项中，Media 选择 RS-232，Stop Bits 选择 1，其他不变。设置完成后，连接控制器。

同样，也要给 PanelView 600 做相应的通信配置，启动 PanelView 600，进入 CONFIGURATION MODE 中的 COMMUNICATION SETUP，然后按表 4-3 设置。

表 4-3 变频器通信设置

参　数	参数名称	设　置
Baud	波特率	19200
Data bits/Parity	数据位数/奇偶校验	8/NONE
Port	接口	RS-232

然后做一个简单的程序，检验它们之间的通信。Micro 830 上的检验程序如图 4-13 所示。

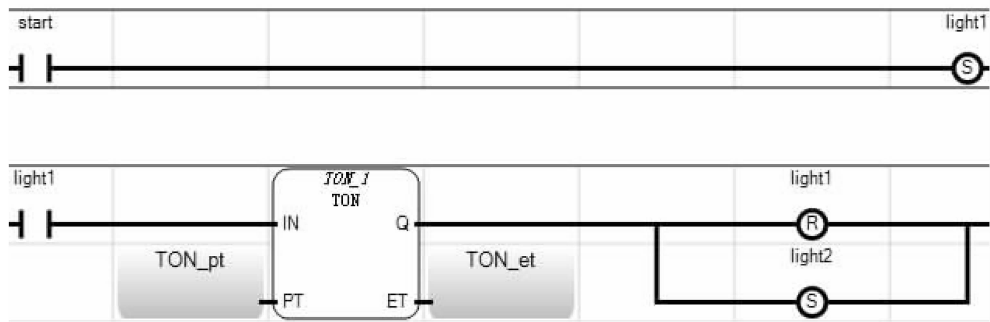


图 4-13　Micro 830 上的检验程序

start 置 1 点亮 light1，从而使能通电延时计时器 timer，计时器开始计时，计时完成后 Q 置 1，light1 灭，light2 导通。

下面做一个相应的画面下载到 PanelView 600 中，打开 PanelBuilder 32 软件，程序进入后出现如图 4-14 所示的创建新应用，选择“Creat a new applica...”。弹出图 4-15 所示对话框，键入应用名称 test，PanelView 版本选择 PV600，协议选择 Modbus，点击“Catalog Revision Numbers”，选择 4. 10-4. xx，这是所使用的 PanelView 600 版本号（见图 4-16）。选择完成后点击“OK”，进入 PanelBuilder 32 主界面。



图 4-14　创建新应用

然后进行通信配置，点击“Application Settings”下的“Communications Setup”，出现图 4-17 所示界面。

这里选择 Baud Rate 为 19200，Data Bits/Parity 为 8/NONE，Port Configuration 为 RS-232。在下面节点名称中键入 traffic，节点地址键入 1，这是在 Modbus 网络中控制器的节点地址，由于 Panel View600 在 Modbus 中为主节点，所以可以不设节点地址，而这里是明确 Panel View600 要通信的控制器在 Modbus 网络中的节点地址，之前 Micro830 控制器的节点地址设为 1，因此这里 Node Address 也设为 1，点击“OK”。

然后做出相应的画面，先做出 start、light1、light2 标签，start 标签赋给一个保持型开关，在 PanelView 600 上使用 F1 控制，light1 和 light2 分别赋给形状如小灯的两个多态指示

器。在这里设置 start 标签置 0 时，开关蓝色且闪烁，置 1 时，黄色不闪烁。light1 置 0 时，小灯 1 为红色且闪烁，置 1 时，黄色不闪烁。light2 与 light1 设置相同。

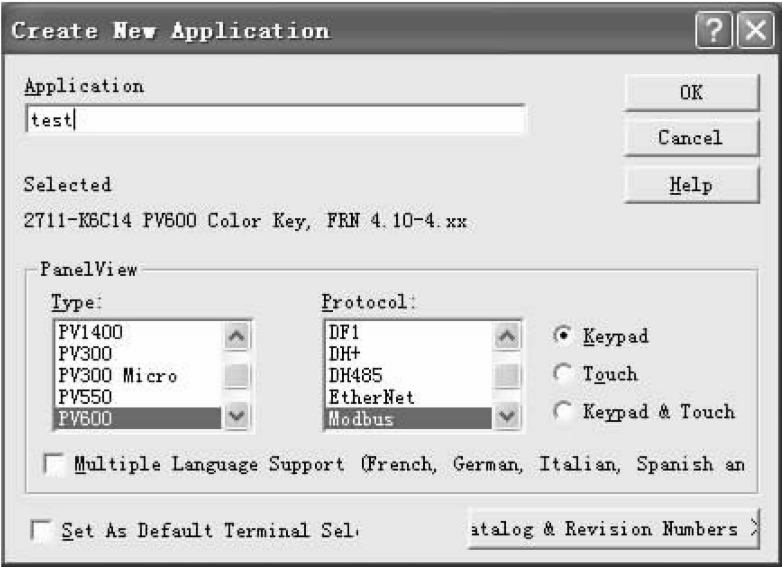


图 4-15　选择型号和通信协议

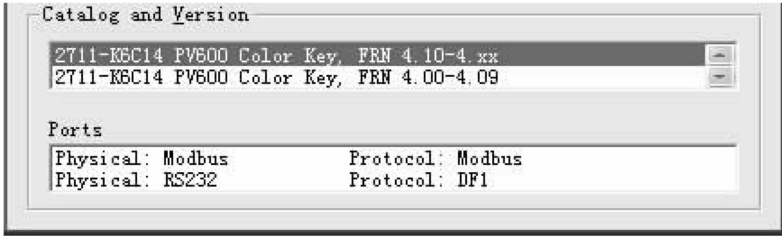


图 4-16　选择版本号

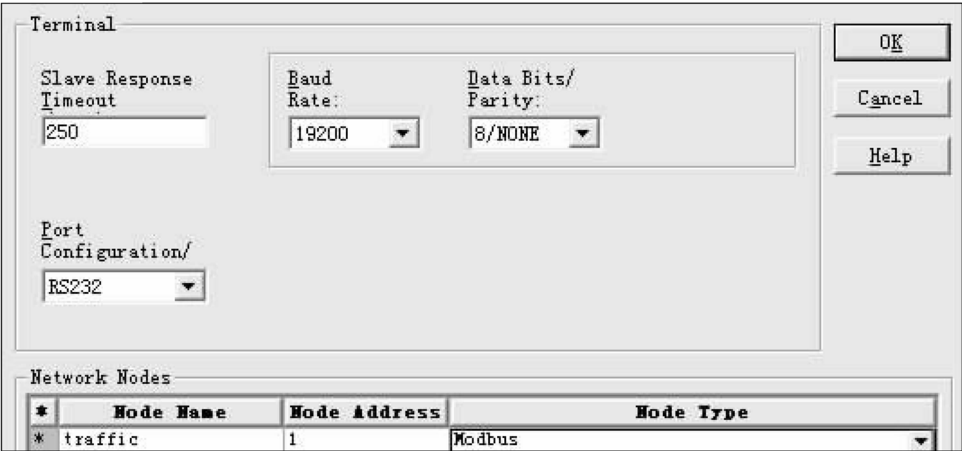


图 4-17　通信配置界面

这里需要了解 Modbus 网络的 Tag 编辑格式，Modbus 数据编址见表 4-4。

表 4-4　Modbus 数据编址

地　址	地址类型	数据类型	备　　注
0xxxxx	Discrete Output Coil（离散输出或者线圈）	位读或者写	通过输出模块来驱动一个实际的输出，或者用来设置一个或者多个内部线圈。一个线圈能被用来驱动多个触点
1xxxxx	Discrete Input Status（离散输入状态）	位只读	用来在逻辑程序中驱动触点。输入模块控制输入状态
3xxxxx	Input Register（输入寄存器）	字只读	从一个外部源保持数字型的输入（例如，一个模拟量信号或者从高速计数器接收的数据）。一个 3x 的寄存器也能存储 16 位离散的信号，这些信号可以存储为位或者 BCD 代码的格式
4xxxxx	Output Holding Register（输出保持寄存器）	字读和写	用来存储数字的信息（十进制或者二进制）或者向输出模块发送信息
6xxxxx	Extended Memory Register（外部存储寄存器）	逻辑程序访问	用来存储外部内存区域的信息。只有 24 位的 CPU 才支持外部存储（例如，984B，E984-785 和 Quantum 系列的 PLC）

在 PanelBuilder32 的 Tag Editor（标签编辑）选项中，定义 PanelView 的标签类型。把 start、light1 和 light2 的标签类型都设置为 Output Coil，地址也相应设置为 000001、000002、000003。标签设置如图 4-18 所示。

Tag Name	Node Name	Type	Address	Initial Valu
start	traffic	Ouput Coil	1	0
light1	traffic	Ouput Coil	2	0
light2	traffic	Ouput Coil	3	0

图 4-18　标签设置

然后还要在 Micro830 控制器中进行地址的映射，进入 CCW 软件，在 Communication Ports 中选择 Modbus Mapping，在弹出的对话框中，按照如图 4-19 所示填写。

Variable Name	Data Type	Address	Addresses Used
start@test	Bool	000001	000001
light1@test	Bool	000002	000002
light2@test	Bool	000003	000003

图 4-19　地址映射

这样，就把控制器中程序的标签与 Panel View600 画面的标签做了映射。

下载程序到控制器，然后进行编译操作，程序处于运行状态。接下来下载画面到 Panel View600 中，在 PanelBuilder 32 中点击“File”下的“Download”，弹出如图 4-20 所示对话框，选择 DF1 Point-to-Point -Internal COM1，点击“OK”，进入下载页面。

画面下载完成后，可以看到 PanelView 600 显示屏上出现所做的画面，接下来进入 CCW

中的程序界面，这时，按 Panel View600 屏上的“F1”，给 start 开关置 1，发现 light1 亮，通电延时计时器使能，开始计时，计时结束后，light2 亮，而 light1 灭。程序和画面显示同步。至此就完成了 Micro830 控制器通过 RS-232 串口和 PanelView 600 的通信。

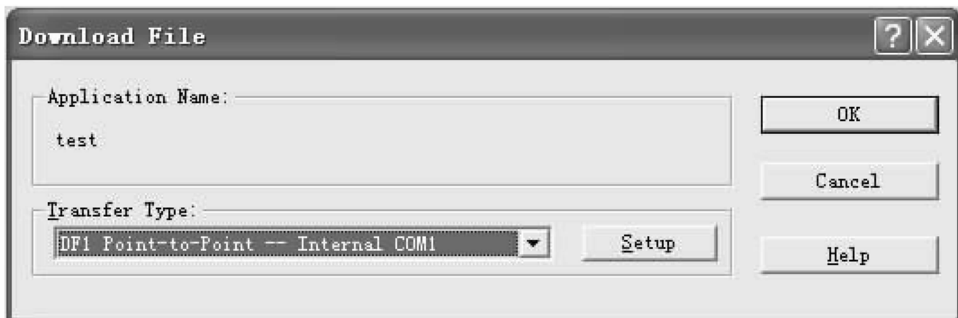


图 4-20　下载接口选择

4.5　Ethernet 通信

Micro800 系列中 Micro850 控制器配有一个 RJ45（8 针水晶头）的以太网端口。本地以太网通信通道允许 Micro850 控制器连接到一个本地网络，为不同的设备提供 10Mbit/s 或 100Mbit/s 的传输速率。

Micro850 控制器支持以下以太网协议：Ethernet/IP 服务、Modbus/TCP 服务、DHCP 客户端。

Micro850 控制器配有两个以太网的指示灯，作用分别为：模块状态，网络状态。以太网组态设置：

1）打开 Connected Components Workbench（CCW）软件（例如 850），在组态设备中选择控制器属性，点击“Ethernet”。

2）在 Ethernet 下点击 Internet Protocol。

设置组态网络的 IP 地址，可以通过 DHCP 自动获取 IP 地址或手动设置 IP 地址、子网掩码和默认网关，如图 4-21 所示。



图 4-21　IP 地址、子网掩码和默认网关

TIP：以太网端口默认以下设置：DHCP（动态 IP 地址）；地址重复检测：开。

3）点击检查框的重复 IP 地址检测使能重复地址检测。

4）在 Ethernet 栏下点击“Port Settings”，如图 4-22 所示。

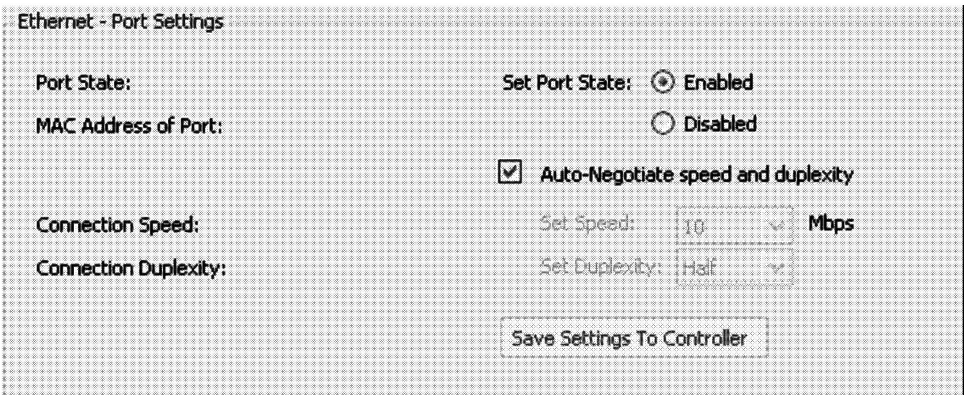


图 4-22　以太网端口设置

5）设置 Port State 为 Enabled 或 Disabled。

6）手动设置连接速度和二重性，不选择 Auto-Negotiate speed and duplexity 选项。然后，设置 Speed 为 10Mbit/s 或 100Mbit/s，Duplexity 为 Half 或 Full。

7）如果想要保存当前设置到控制器中，点击保存设置。

8. 在设备组态栏中，Ethernet 栏下点击“Port Diagnostics”，当在调试模式下就可以观测实时更新的计数值。

4.6　小结

本章以 Micro830 为例，详细介绍了 Micro800 系列控制器的 USB 通信、RS-485 通信和 RS-232 串口通信、以太网通信。

Micro800 系列控制器通过 USB 通信与计算机连接，用于下载和上载程序。RS-485 通信是通过嵌入模块 2080-serialisol 或使用 1761-NET-AIC 接口转换模块来实现。连接后，Micro800 系列控制器可以通过 Modbus 协议与其他通信设备通信。RS-232 串口通信使 Micro800 系列控制器与其他设备进行点对点通信，双方遵从半双工的 Modbus 协议。Micro850 的嵌入式 EtherNet/IP 通信可用于连接 PC、PanelView Component HMI、Kinetix 伺服和 PowerFlex 变频器。

习　题

1. 简述 Micro800 控制器的网络结构。
2. Micro850 以太网的 IP 地址如何分配？
3. Micro830 控制器的网络通信方式有哪些？
4. Micro850 控制器的网络通信方式有哪些？

思考题

1. 相距甚远的两台电动机 A 和 B，A 起动 10s 后 B 起动，试用多种方法实现（控制器数量、网络类型均不限）。

2. 比较 USB、RS-232、RS-485、Ethernet 通信方式的优缺点。

3. Micro800 控制器分别通过哪些模块连接到各种网络上？

4. 除了本章介绍的网络类型，你是否还了解其他工业网络？试分析它们各自的特点。

第 5 章

Micro800 控制器的编程示例

学习目标

- 熟悉 Micro800 控制器编程软件 CCW 的安装
- 了解 CCW 编程软件的编程环境
- 掌握 CCW 编程软件的编程步骤
- 掌握 CCW 中程序上载、下载和调试

5.1　编程软件的安装

Micro800 控制器的编程软件 CCW 已经在 Microsoft Windows XP、Microsoft Windows Vista 和 Microsoft Windows7 系统上测试成功，对系统的具体要求见表 5-1。

表 5-1　CCW 软件对计算机操作系统的要求

操作系统	要　求
Microsoft Windows XP	Microsoft Windows XP Professional（32-bit）with Service Pack 3 这个版本的 CCW 软件预期能够在其他所有的微软 Windows XP 系统上工作（不含 Windows XP 系统的家庭版）
Microsoft Windows Vista	Microsoft Vista Business（32-bit）with Service Pack 2. Microsoft Vista Home Basic（32-bit）with Service Pack 2.
Microsoft Windows 7	Microsoft Windows 7 Home Basic（32-bit）. Microsoft Windows 7 Home Premium（32-bit）.

下面介绍 Micro800 控制器编程软件 CCW 的安装过程。在安装之前，请确保计算机系统满足表 5-1 的要求。安装步骤如下：

1）打开安装文件，找到如图 5-1 所示图标，双击即可。



图 5-1　编程软件安装图标（开发版）

2）双击安装图标后，弹出的对话框提供可以选择所使用的语言。

3）点击“继续”，弹出选择版本的对话框，选择典型版本，点击“下一步”。

4）在弹出的对话框输入用户信息（用户名：Rockwell Automation，Inc；组织：Rockwell Automation，Inc），点击“下一步”。

5）在弹出的对话框中选择“接受”，点击“下一步”。

6）在弹出的对话框中点击“安装”，如果需要可以更改安装路径。

安装完成后，会提示安装完成，点击“完成”即可。

5.2　编程软件简介

软件 CCW 是 Micro800 系列控制器的程序开发软件，在这个软件中，不仅可以组态 Micro800 控制器，还可以组态触摸屏和变频器。下面介绍软件的使用方法。

5.2.1　创建一个新项目

本节主要介绍怎样用 CCW 软件创建一个新的项目。

1）打开 CCW 软件，按照下面所示的路径：开始→所有程序→Rockwell Automation→CCW→Connected Components Workbench。

软件打开后，显示的是一个新建项目的界面，整个界面可以分为三部分，最左边是项目

组织器窗口，在项目组织器窗口中显示建立项目所选择的控制器及其项目中建立的变量和编写的程序，并可以对控制器及其程序和变量进行编译、删除等操作。中间为工作区和输出窗口，在工作区中显示编写的程序或者要组态的控制器，输出窗口中可显示编译程序后的提示信息。最右边为设备工具箱和指令工具箱，上面为设备工具箱，这里的设备包括控制器、变频器和触摸屏三部分，打开相应的菜单可以选择相应的设备；下面为指令工具箱，当为建立的项目选择了控制器以后，编写程序时这里会显示要用的指令，只需把指令拖拽到工作区的编程区域就可以使用。

2）要建立一个项目，首先要打开左边的设备工具箱，并打开其中的控制器菜单，选择所需要的控制器型号，这里选择 2080-LC30-24QWB。

3）在选定的控制器上双击或者是直接把控制器拖拽到项目组织器，就会在项目组织器上出现如图 5-2 所示的窗口，这样就创建了一个基于 Micro830 控制器的项目。

4）该项目的名称显示在项目组织器窗口的名称字段中。

5）点击“保存”按钮，就可保存项目，新建的项目默认保存在 C：\ Documents and Settings \ Administrator \ My Documents \ CCW 文件夹下。

这样就完成了一个新项目的创建。

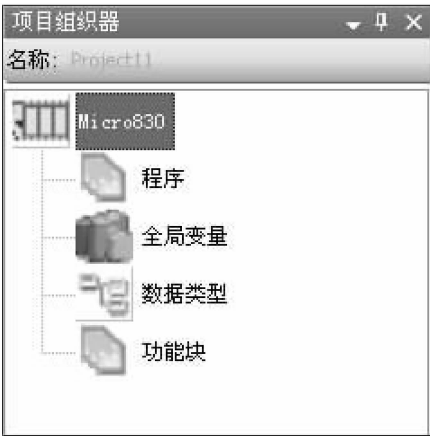


图 5-2　项目组织器窗口

5.2.2　组态控制器嵌入模块

在 CCW 中可以很直观地组态 Micro800 控制器的嵌入模块。打开一个项目，在窗口左边的项目组织器窗口中双击控制器图标，可以打开组态嵌入模块的界面。本例中所选的控制器是 Micro830 控制器，型号为 2080-LC30-24QWB，共可以容纳 3 个扩展模块。在控制器的空白的槽上右键单击，弹出如图 5-3 所示的模块选项，用户可以根据实际情况，选择所需模块的型号。这里假设这 3 个空白槽上分别插模拟量输入模块 2080-IF4、模拟量输出模块 2080-OF2 和通信接口模块 2080-SERIALISOL。在第一个空白的槽上右键单击，选择 2080-IF4 单击即可把模块嵌入到控制器中。用同样的方法可以嵌入其他两个模块。

嵌入模块后，在控制器下方的工作区内可以对每个嵌入模块进行组态，图 5-4 所示为 2080-IF4 模块的组态界面，这里可以对模块每个通道的输入类型、频率和输入状态进行组态。同

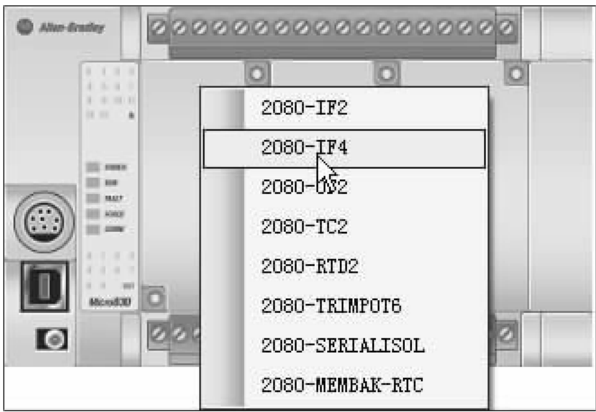


图 5-3　选择嵌入模块

时在这里也可以组态控制器自身所带的通信端口和 I/O 点。



图 5-4　2080-IF4 模块的组态界面

5.2.3　创建一个新程序

完成了系统的硬件组态以后，就可以编写程序文件了。首先要创建一个新程序，在项目组织器窗口中用鼠标右键单击控制器图标，选择添加一个新的梯形图程序，如图 5-5 所示。

程序新建完成后，如图 5-6 所示，然后用鼠标右键单击程序，选择对程序重新命名。



图 5-5　新建梯形图程序

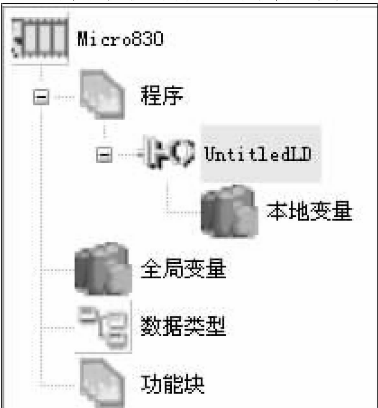


图 5-6　新建的梯形图程序

把程序命名为 test，创建的程序将完成以下功能：有两盏灯 light1 和 light2，在第一盏灯亮 2s 以后，熄灭第一盏灯，点亮第二盏灯。则首先要创建编写程序所需要的变量，分别有 start、light1、light2 和计时器 timer。程序中所用到的变量可以是全局变量，也可以是本地变量，在项目组织器窗口中打开本地变量或者全局变量，只要双击其图标即可。这里采用本地变量，打开本地变量列表，建立程序所需要的变量，如图 5-7 所示。

test-VAR　Micro830

名称	数据类型	维度
start	BOOL	
light1	BOOL	
light2	BOOL	
timer	TON	
timer.IN	BOOL	
timer.PT	TIME	
timer.Q	BOOL	
timer.ET	TIME	
timer.Pdate	TIME	
timer.Redge	BOOL	

图 5-7　建立程序所需要的变量

这里除了要建立本次编程所需的变量以外，介绍一下在项目中怎样建立数组。要建立数组首先要在 CCW 的项目组织器窗口中找到 Data Types，打开后建立一个数组的类型。如图 5-8 所示，建立数组类型的名称为 a，数据类型为布尔型，建立一维数组，数据个数为 10（维度一栏写 1..10），打开全局变量列表，建立名为 ttt 的数组，数据类型选择为 a，如图 5-9 所示。同理，建立二维数组类型时，维度一栏写 1..10..10。

数组　已定义的字

名称	数据类型	维度
MODBUSLOCADDR	WORD	1..125
ASCIILOCADDR	BYTE	1..82

图 5-8　定义数组的数据类型

ttt　BOOL
WORD
a
ASCIILOCADDR
MODBUSLOCADDR
ABL
ACB

图 5-9　建立数组

下面继续介绍程序的建立。在项目组织器窗口中双击程序图标，打开编程窗口，在工具栏中拖拽所需要的指令到编程梯级。把常开指令拖拽到梯级上以后，会自动弹出变量列表，编程人员可以直接选择需要的变量，如图 5-10 所示，这里选择启动按钮。然后以同样的方法，把第一个梯级完成，如图 5-11 所示。添加一个新的梯级，开始编写第二个梯级。在第二个梯级中需要用到计时器，这里计时器创建时选择功能块指令，把功能块指令拖拽到梯级上以后，会自动弹出选择功能块的对话框，选择 TON 功能块，选择完成后，计时器的名字在“范例”项中选择，选择前面建立的计时器“timer”。为计时器定时 2s，在计时器的 PT 输入处双击，输入 T#2s 即可。熄灭第一盏灯的同时，点亮第二盏灯，则梯级需要一个分支，从工具栏中拖拽梯级分支到计时器后面的梯级上，然后添加复位线圈和置位线圈，编好的梯级如图 5-12 所示。



图 5-10　选择所需要的变量

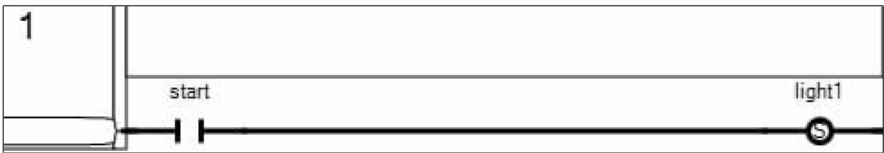


图 5-11　点亮第一盏灯的梯级

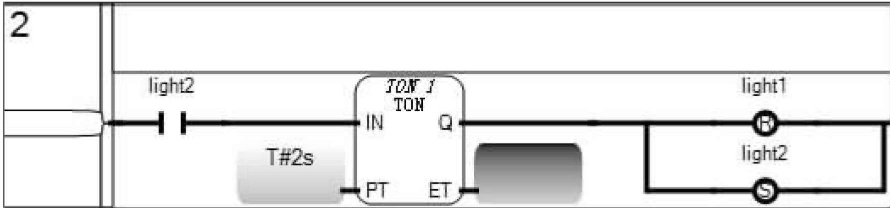


图 5-12　第二个梯级

通过以上步骤就完成了程序的编写，右键单击程序图标，选择生成，如图 5-13 所示，对程序进行编译，编译无误后会提示编译完成。

5.2.4　组态变频器



图 5-13　编译程序

CCW 的一个显著的优点就是不仅可以组态控制器，编写控制器程序，还可以组态变频器和触摸屏，很大地方便了编程人员。下面来介绍变频器的组态。

首先要把变频器和计算机连接起来，这里用 22-SCM-232 模块，该模块可以将变频器上的 RS-485 通信接口转换成串口连接到计算机上。打开 RSLinx 软件，然后打开组态驱动对话框，选择建立 RS-232 DF1devices 驱动，如图 5-14 所示，点击添加新的驱动，就会

打开如图 5-15 所示的组态 DF1 驱动的对话框，通信端口选择变频器连接的计算机串口，这里选择 COM1，通信波特率选择 9600，因为 22-SCM-232 模块的默认波特率为 9600，所以这里波特率要和模块的保持一致，其他选项默认即可。设置完成后，不要选择自动组态，直接点击“OK”即可，然后打开 RSLinx 软件的 RSWho 窗口，如图 5-16 所示，表示模块和计算机已经成功地连接了。

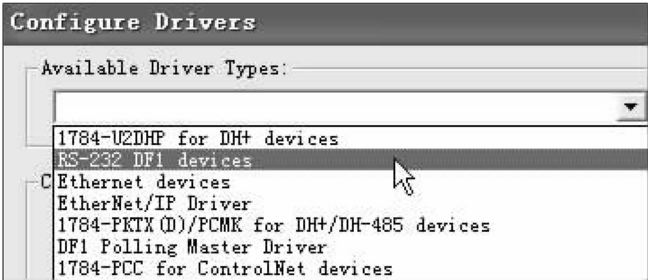


图 5-14　添加 DF1 驱动

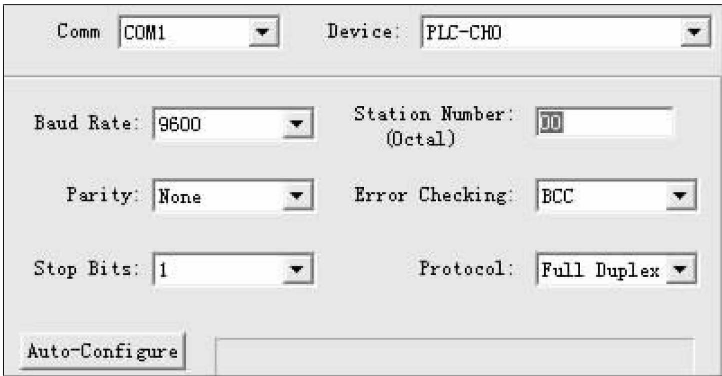


图 5-15　组态 DF1 驱动的对话框

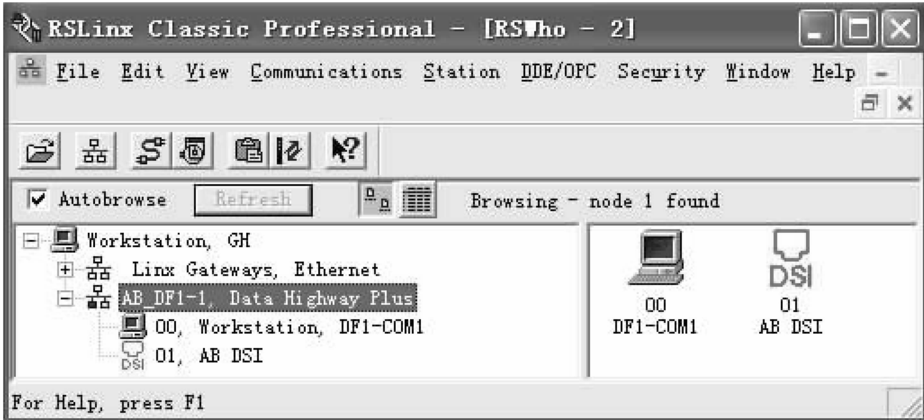


图 5-16　已连接的变频器组态

连接完成后，打开 CCW 的一个工程，在设备工具箱中打开驱动器文件夹，选择 PowerFlex40 变频器（这里以 PowerFlex40 变频器为例介绍如何在 CCW 中组态变频器）。把变频器拖拽到项目组织器中，如图 5-17 所示，双击变频器图标，可以打开变频器的组态界面，界面中的连接按钮用来连接实际的变频器，上传和下载按钮用来上传和下载变频器的参数配置，参数和属性按钮用来配置变频器的参数和属性，向导按钮是为新手提供的快速配置变频器各项参数的按钮。

变频器的组态界面中左下角的加（+）按钮用来给变频器添加外围设备，点击加（+）按钮，如图 5-18 所示，选择这里用到的模块 22-SCM-232，点击添加按钮，添加完成后，可以配置模块的各项参数和属性。下面先连接到实际设备，然后再讲述变频器各项参数的配置。点击“连接”按钮，在弹出对话框中选择要连接的变频器，点击“确定”即可。连接完成后，组态界面变为如图 5-19 所示的界面，并且连接的同时会上传当前所连接设备的配置信息。



图 5-17　在项目组织器中添加 PowerFlex40 变频器



图 5-18　添加变频器外围设备

从图 5-19 中可以看到下载按钮不能使用了，这是因为要下载配置好的变频器参数不能连接设备，要先断开连接，然后点击“下载”按钮，再选择下载到哪个变频器。



图 5-19　在线的变频器组态界面

点击“参数”按钮，会打开变频器的参数对话框，参数列表如图 5-20 所示，在这里用户可以组态变频器的所有参数。但由于通常不是所有的参数都要组态，推荐新用户使用启动向导来配置变频器参数。

点击“属性”按钮，可以看到变频器的各项属性，在离线的情况下，用户还可以修改变频器的一些属性，包括变频器的版本和端口等。

参数列表 - PowerFlex 40_1*

#	名称	值	单位	内部值	默认
28	Stp Logic Status	0		0	0
29	Torque Current	0.00	A	0	0.00
30	Reserved	0		0	0
31	Motor NP Volts	380	V	380	460
32	Motor NP Hertz	50	Hz	50	60
33	Motor OL Current	1.1	A	11	4.0
34	Minimum Freq	0.0	Hz	0	0.0
35	Maximum Freq	50	Hz	50	60
36	Start Source	Comm Port		5	Keypad
37	Stop Mode	Ramp, CF		0	Ramp, CF
38	Speed Reference	Comm Port		5	Drive Pot
39	Accel Time 1	2.0	Sec	20	10.0
40	Decel Time 1	2.0	Sec	20	10.0
41	Reset To Defalts	Ready/Idle		0	Ready/Idle

图 5-20　变频器参数列表

点击“故障”按钮，会显示变频器的故障列表，如图 5-21 所示。在这里可以看到变频器所出现的故障，并可以在故障消除后对故障进行清除。

重置按钮用来重置变频器的各项参数，重置的过程中系统会提示一些信息。这里不再演示重置方法。

下面来讲述在线的情况下如何使用启动向导来配置变频器参数。点击向导按钮，打开如图 5-22 所示的对话框，选择 PowerFlex40 启动向导，点击“选择”按钮，打开向导欢迎对话框，该对话框是用来介绍向导并给出使用向导的提示和技巧的。按照向导的提示，设置变频器参数。

故障 - PowerFlex 40_1*

代码	描述
7	Motor Overload
4	UnderVoltage
81	Comm Loss

清除故障

清空故障队列

状态

Stopped

图 5-21　变频器故障列表

启动向导设置变频器的步骤如下所示：

1. 重置参数

点击“下一步”，弹出重置参数的对话框，这里点击“重置参数”按钮，可以把变频器的各项参数还原为默认值。如果不需要重置参数，直接点击“下一步”即可。

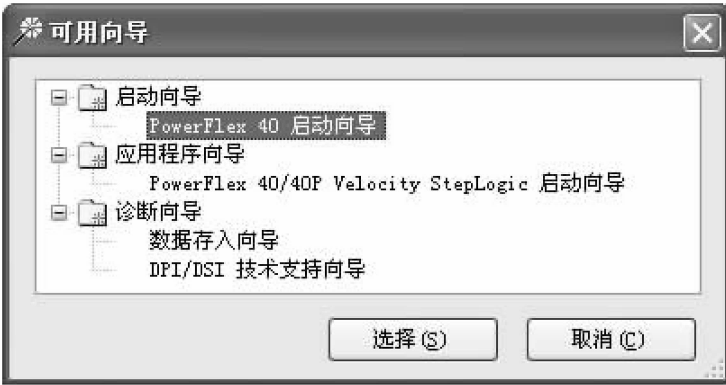


图 5-22　启动变频器配置向导

2. 电机控制

点击“下一步”来设置电机控制，如图 5-23 所示。在这里可以设置电机的力矩特性模式，共有两个选项（V/Hz 和 Sensrls Vect），同时还可以设置 FLA 上 Slip 赫兹、提升选择、启动提升、截断电压、截断频率和最高电压等。



图 5-23　电机控制

3. 电机数据

点击“下一步”，设置电机数据，如图 5-24 所示。这里用户要根据变频器所控制的电机的铭牌来设置电机的电流、电压和频率。本次实验用到的电机的额定电流为 1.12A，额定电压为 380V，额定频率为 50Hz。



图 5-24　电机数据

4. 停止模式/制动模式

点击“下一步”，设置电机的停止模式/制动模式，如图 5-25 所示。这里要设置的参数有两项（DB 电阻器选择和停止模式），点击下拉菜单，电阻器选择有以下几种：Disabled、Norma RA Res、No-Protection 和 3% DutyCycle ~ 99% Dutycle；停止模式的选择有：Ramp CF、Coast CF、DC Brake CF、Ramp、Coast、DC Brake、DC BrakeAuto、Ramp + EM B CF 和 Ramp + EM Brk。

5. 方向测试

点击“下一步”，进行方向测试，如图 5-26 所示。这个测试只有在在线的情况下才能进

行。在参考值处给电机一个速度，这里设为 10Hz，然后点击绿色的启动按钮，电机正转，查看电机的转动方向是否为正转，如果是正转点击“停止”按钮，然后点击 Jog 按钮，按住 Jog 键，电机转动，松开 Jog 键电机停止转动。完成这些测试以后，如果电机转动方向正确，在问题“应用程序的电机旋转方向是否正确”的下面，选择“是”，然后就会提示测试通过。



图 5-25　停止模式/制动模式

6. 自动调谐

点击“下一步”，进行自动调谐，这一步也只能在在线的情况下进行。通过自动调谐，启动器可以对电机特性进行取样，并正确设置其 IR 压降和磁通电流参考，仅当电机卸除负荷时，才能使用旋转调节。否则，请使用静态调节。调节完成后，系统会提示测试已完成。



图 5-26　方向测试

7. 斜率/速度限制

点击“下一步”，设置斜率/速度限制，如图 5-27 所示。这里可以设置最大频率和最小频率，还可以设置是否启动反向操作。这是因为在有些系统中，电机是不能反转的，例如传送带等。

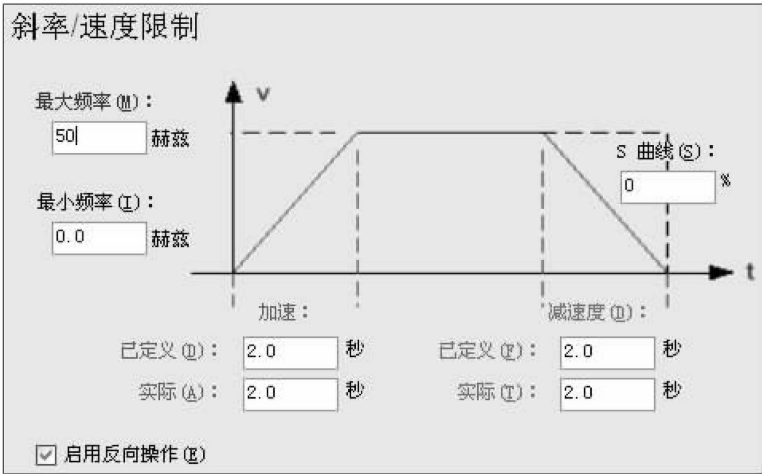


图 5-27　斜率/速度限制

8. 速度控制

点击“下一步”，设置速度控制，如图 5-28 所示。这里设置速度参考的来源，点开下拉菜单，有以下选择：Drive Pot、InternalFreq、0-10V Input、4-20mA Input、Preset Freq、Comm Port、Stp Logic 和 Anlg In Mult。这里选择 Comm Port（通信端口）。



图 5-28　速度控制

9. 数字输入

点击“下一步”，配置数字输入参数，如图 5-29 所示。这里可以设置停止源、启动源、数字输入 1 ~ 4 和预置频率 0 ~ 7。用户可以根据需要来设置启动源、停止源、数字输入和预置频率。

10. 继电器输出

点击“下一步”，设置继电器输出，如图 5-30 所示。点开继电器输出的下拉菜单，选择适合的继电器输出。



图 5-29　数字输入参数

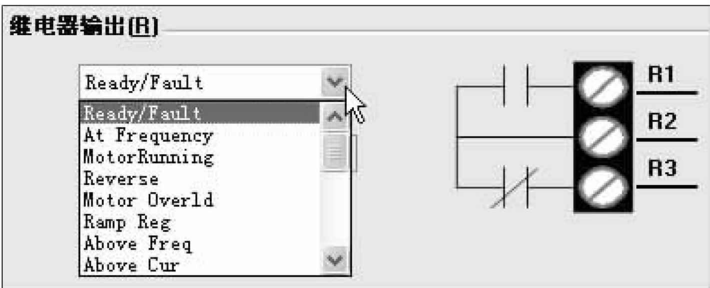


图 5-30　继电器输出

11. 光电耦合输出端

点击“下一步”，设置光电耦合输出端，如图 5-31 所示。这里可以设置光电耦合输出端的逻辑、光电耦合输出端 1 和光电耦合输出端 2。

12. 模拟输出

点击“下一步”，设置模拟输出，如图 5-32 所示。

13. 待定更改

完成了上面的设置，点击“下一步”，显示待定修改界面，这里提示用户以上做了哪些修改，点击“完成”即可把所有的修改应用到变频器中。

启动向导提供了一种清晰的思路来快速组态变频器，它虽然没有覆盖到每一个参数，但大多数应用项目中常用的必须配置参数都已包括在内。

以上参数的配置都是在变频器在线的情况下进行的。断开在线的变频器，也可以用向导来配置变频器，只是有些步骤不能完成，但这不妨碍对变频器基本参数的配置。离线变频器参数的配置步骤和在线变频器参数配置的步骤一样，这里就不再赘述了。下面来介绍一下变频器参数的下载。



图 5-31　光电耦合输出端

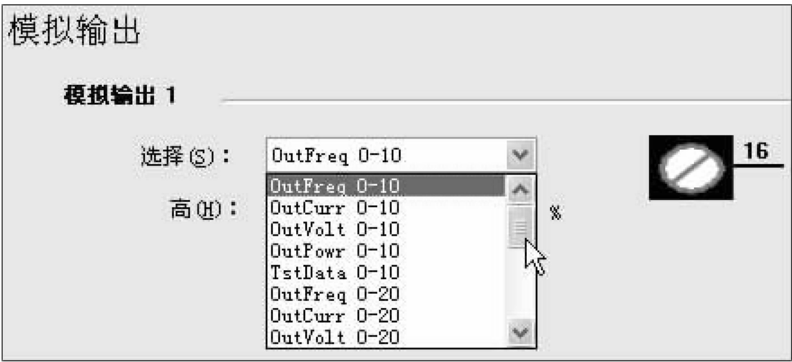


图 5-32　模拟输出

点击“下载”按钮，会弹出下载对话框，点击对话框上的更改按钮，更改下载路径，会弹出如图 5-33 所示的对话框，选择目标变频器，点击“确定”后，选择下载到整个设备，就会把所做的修改下载到变频器，下载完成后，会弹出下载成功的提示框，点击“确定”后，所有的修改都会应用到变频器上。上传变频器参数的过程和下载的基本一致，上传过程中，系统还会提示一些信息，这里就不再赘述了。

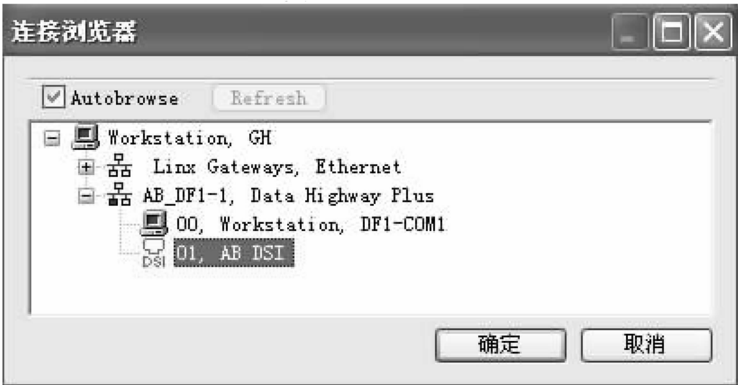


图 5-33　选择目标变频器

5.3　编程示例

本节将以交通信号灯为例来具体介绍功能块的编写方法及其在主程序中的使用方法。

5.3.1　自定义功能块示例

Micro800 控制器的一个突出的特点就是在用梯形图语言编写程序的过程中，对于经常重复使用的功能可以编写成功能块，等需要重复使用的时候直接调用功能块即可，无需重复编写程序。这样就给程序开发人员提供了极大的便利，节省时间的同时也节省了精力。功能块的编写步骤与编写主程序的步骤基本一致，下面简单介绍一下。

在项目组织器中，选择功能块图标，右键单击，选择新建梯形图。新建功能块的名字默认为 UntitledLD，右键单击，选择重命名，可以给功能块取相应的名字。双击打开功能块后可以编写完成功能块的功能所需要的程序，功能块的下面为变量列表，这里的变量为本地变量。只能在当前功能块中使用。

这样就完成了一个功能块程序的建立，然后在功能块中编写所要实现的功能。完成后功能块可以在主程序中直接使用。下面以交通灯功能块为例具体介绍功能块的编程。

要编写的交通灯功能块要完成的功能是：当一个方向的汽车等红灯等了至少 5s 的时候，把另一个方向的绿灯变为黄灯 2s，然后变成红灯，同时前面红灯方向的红灯变为绿灯。

首先把新建功能块命名为交通灯控制功能块（TRAFFIC _ CONTROLLER _ FB），如图 5-34 所示。

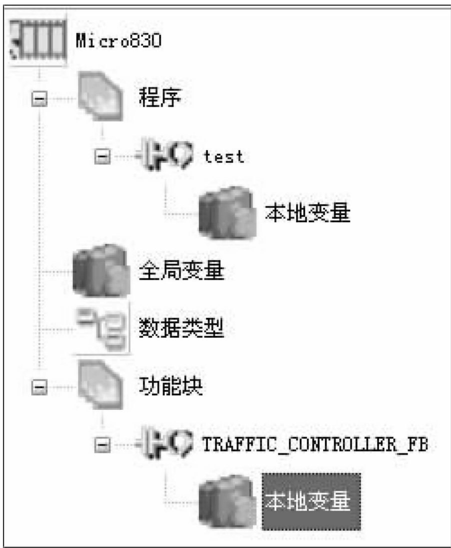


图 5-34　新建交通灯控制功能块

创建一个新的功能块，首先要确定完成此功

能块所需要的输入和输出变量。这些输入/输出变量在项目组织器中的本地变量中创建，如图 5-34 所示，在新建功能块的下面，双击本地变量图标，打开如图 5-35 所示的创建变量的界面。

名称	数据类型	方向	维度	别名	初始值	特性

图 5-35　创建变量的界面

在表格的上部用鼠标右键单击，显示如图 5-36 所示的选项，这里可以对表格列的显示进行重置，默认显示一些常用选项。



图 5-36　对变量表格重置

对于此次要编写的交通灯控制功能块，需要 4 个布尔量输入，分别是 4 个方向的信号，6 个布尔量输出，分别是在东西向和南北向的红、黄、绿交通信号灯。输入/输出的定义是在“方向”一列中定义的，输入用 VarInput 表示，输出用 VarOutput 表示。下面首先来定义此功能块所需要的变量，创建功能块变量如图 5-37 所示。在表格的“名称”一列中输入变量的名字，并设定变量为输入或者输出变量即可，要新建变量，在已经建立的变量处回车即可创建下一个变量。图 5-37 中是完成此功能块所需要的输入和输出变量，一定要在“方向”一列中定义变量为输入或者输出变量，否则在主程序使用此功能块的时候将无法显示其输入/输出变量。在变量列表中除了定义变量的数据类型和变量类型以外，还可以对变量进行别名、加注释、改变维度、设置初始值等操作。

TRAFFIC_CONTROLLER_FB-VAR

名称	数据类型	方向	维度	别名	初始值	特性
N_CAR_SENSOR	BOOL	VarInput				读取
S_CAR_SENSOR	BOOL	VarInput				读取
E_CAR_SENSOR	BOOL	VarInput				读取
W_CAR_SENSOR	BOOL	VarInput				读取
NS_RED_LIGHTS	BOOL	VarOutput				写入
NS_YELLOW_LIGHTS	BOOL	VarOutput				写入
NS_GREEN_LIGHTS	BOOL	VarOutput				写入
EW_RED_LIGHTS	BOOL	VarOutput				写入
EW_YELLOW_LIGHTS	BOOL	VarOutput				写入
EW_GREEN_LIGTHS	BOOL	VarOutput				写入

图 5-37　创建功能块变量

定义了输入/输出变量就可以编写功能块程序了。双击交通灯控制功能块（TRAFFIC _ CONTROLLER _ FB）图标，可打开编程界面。

根据要求可知第一个梯级实现如下功能：如果南北红灯和东西绿灯亮，并且南北向的车等了至少 5s，那么就把东西绿灯变为黄灯。

点击设备工具箱窗口下部的工具箱，展开梯形图工具箱。工具箱里有编写梯形图程序所需要的基本指令，用户只需选择要用的指令，直接拖拽到编程界面中的梯级上即可。

把指令拖拽到梯级上以后，就会自动地弹出变量列表，编程人员可以直接给指令选择所用的变量，这里选择接触器位指令，并添加 NS _ RED _ LIGHTS 变量。用同样的方法添加第二个接触器位指令，变量选择 EW _ GREEN _ LIGHTS，然后选择一个梯形图分支指令，并在上面分别放接触器位指令，变量为 N _ CAR _ SENSOR 和 S _ CAR _ SENSOR。然后添加一个功能块，并选择计时器指令（TON），并给计时器定时 5s。在梯级的最后再添加一个梯级分支，分别放置位线圈 EW _ GREEN _ LIGHTS 和复位线圈 EW _ YELLOW _ LIGHTS。这样就完成了第一个梯级的编写，其功能是：当南北红灯和东西绿灯同时点亮，并且南北车辆等候至少 5s 的时候，复位东西绿灯，同时点亮东西黄灯。

这样就完成了第一个梯级的编写，编写好的梯级如图 5-38 所示。可以在梯级的上方为梯级添加描述信息，也可以在描述处单击右键，选择不显示描述，如图 5-39 所示。这里还可以对梯级或者指令进行复制、粘贴、改变布局等，打开属性对话框还可以设置对象的各种属性，同时还可以打开交叉引用浏览器来查看一个变量在程序中多处使用的情况。

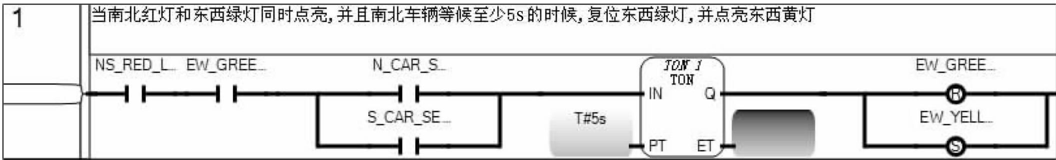


图 5-38　交通灯功能块第一个梯级

根据分析，第二条梯级实现以下功能：当东西黄灯亮 2s 以后，复位东西黄灯和南北红灯，同时置位南北绿灯和东西红灯。其第二个梯级如图 5-40 所示。

经分析可知第三个和第四个梯级与第一个和第二个梯级完成的功能相同，只是方向不同，所以只需把第一个和第二个梯级复制，然后改变一下变量即可，第三个和第四个梯级如图 5-41 所示。

在完成了交通灯功能块的功能以后，还需要加另外的一个梯级用来初始化。当程序第一次被下载到控制器并运行的时候，所有交通信号灯的状态都应该是灭的。最后这个梯级就是用来确保这一点，并同时点亮南北红灯和东西绿灯。交通灯功能块初始化梯级如图 5-42 所示。

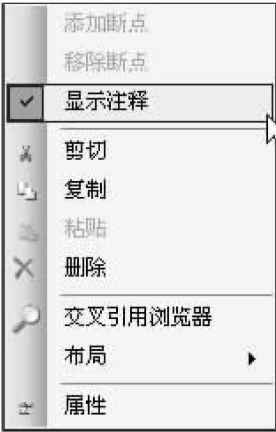


图 5-39　选择梯级描述是否显示

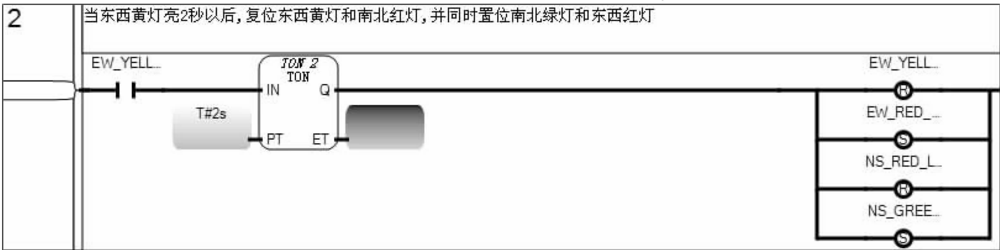


图 5-40　交通灯功能块第二个梯级

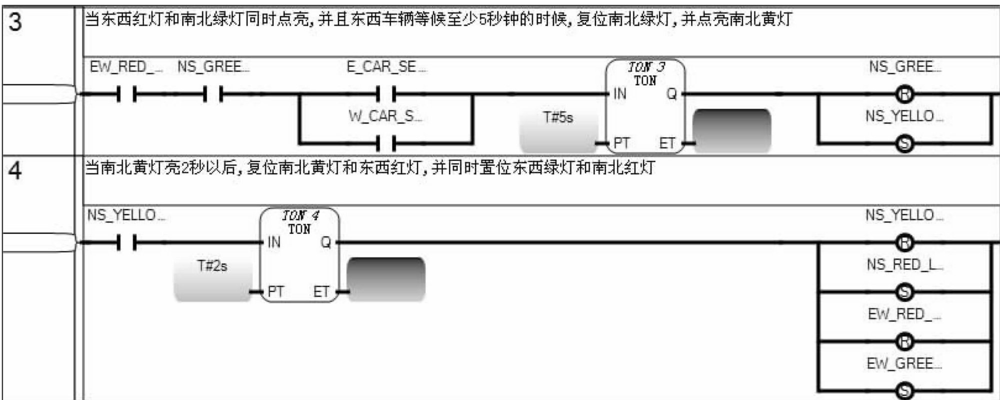


图 5-41　交通灯功能块第三个和第四个梯级

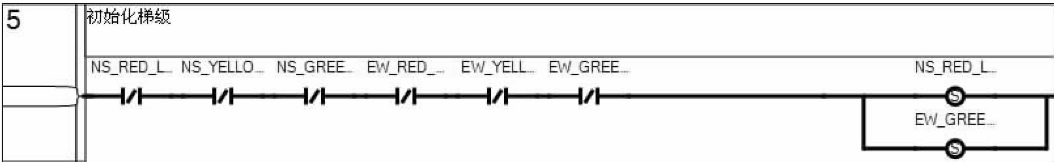


图 5-42　交通灯功能块初始化梯级

到此就完成了交通灯功能块的编写，在项目组织器中，右键单击功能块图标，选择编译（生成），可以对编好的程序进行编译，如果程序没有错误，点击“保存”按钮即可保存。如果程序中出现错误，在输出窗口中将出现提示信息，提示程序编译出现错误，同时会弹出

错误列表，如图 5-43 所示，在错误列表中会指出错误在程序中的位置。双击错误信息行，可以跳转到程序的错误位置，对错误的程序做出修改。然后再次对程序进行编译，程序编译无误后点击“保存”按钮即可。



图 5-43　错误列表

5.3.2　自定义功能块的使用

上节完成了对交通灯功能块的编写，本节介绍编写的交通灯功能块在主程序中的使用。

1）首先要在项目组织器窗口中创建一个梯形图程序，右键单击程序图标，选择新建梯形图程序。

2）创建新程序以后，将程序重新命名为交通灯控制（Traffic _ Light _ Control）。

3）双击交通灯控制图标，打开编程界面，在工具箱里选择功能块指令拖拽到程序梯级中，如图 5-44 所示。拖拽功能块指令到梯级以后，会自动弹出功能块选择列表，找到编写好的交通灯控制功能块，选择即可。

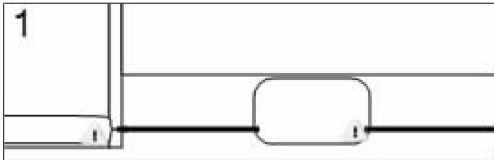


图 5-44　编写主程序

如图 5-45 所示选择自己编写的交通灯功能块，在下面的参数列表中可以看到交通灯功能块中所有的输入和输出参数，在该参数列表中可以对这些参数进行必要的设置，完成参数的设置后，把图 5-45 中右下角的两个复选框选上，第一个显示参数表示在使用功能块时显示功能块的所有输入和输出参数，第二个 EN/ENO 表示使能功能块的输入和输出。如果这里不选择，将无法在主程序中使用功能块。

完成参数设置以后，单击“OK”，交通灯功能块将出现在程序中。可以看到交通灯功能块有 4 个输入变量和 6 个输出变量，单击输入或者输出，可以出现选择变量的下拉菜单，如图 5-46 所示，在此下拉菜单中为功能块的输入/输出选择合适的变量。

由于变量默认的名字太长，为了方便起见，可以对使用的变量别名，在功能块的第一个输入处双击，可以打开变量列表，在此列表中可以对变量进行别名，全局变量列表如图 5-47 所示。



图 5-45　设置交通灯功能块

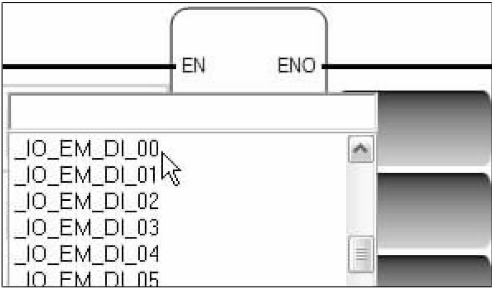


图 5-46　为功能块输入/输出选择变量

_IO_EM_DI_00	BOOL		DI0		Read
_IO_EM_DI_01	BOOL		DI1		Read
_IO_EM_DI_02	BOOL		DI2		Read
_IO_EM_DI_03	BOOL		DI3		Read
_IO_EM_DI_04	BOOL		DI4		Read

图 5-47　全局变量列表

对变量别名以后，变量的别名将出现在功能块上，如图 5-48 所示。

这样就完成了程序的编写，在项目组织器窗口中右键单击交通灯控制图标，选择编译（生成），对主程序编译。编译完成后点击“保存”即可。

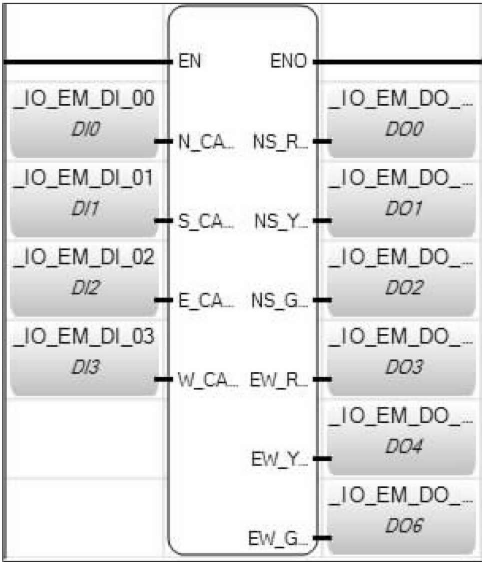


图 5-48　别名后的变量

5.3.3　程序的导入和导出

当有多个项目需要使用相同的功能时，为了避免重复工作，编程人员可以把现有的程序从项目中导出，然后再导入到另外的项目中。下面介绍程序的导入/导出方法。

1. 程序导出

在项目组织器窗口中，选择已经建立的程序，右键单击，选择导出，如图 5-49 所示。

选择导出程序后，弹出如图 5-50 所示的窗口，这里可以选择只导出变量，也可以选择全部导出，还可以对导出的文件加密。这里我们选择全部导出，并对文件加密。然后点击下面的导出按钮，在弹出的对话框中可以改变导出文件的路径和名字。这里把导出文件保存到桌面，并命名为“Traffic _ Light _ Control”。



图 5-49　导出程序

导出文件成功后，会在软件工作区的输出窗口中显示消息，提示导出完成，并写明了导出文件的位置和名字。

2. 程序导入

下面把导出的程序导入到一个新的项目中。首先打开一个新的项目，在程序图标处右键单击，选择导入，类似导出操作。选择后弹出如图 5-51 所示的导入程序窗口，在窗口中可以选择要导入的内容，可以只导入主程序或者只导入功能块程序，也可以全部导入，单击“浏览”按钮，选择要导入的文件，选择打开。然后点击导入文件窗口下方的“导入”按

钮，就可以导入文件了。由于在导出文件的时候对文件设定了密码，所以在点击“浏览”按钮，选择要导入的文件时需要输入文件密码，在密码输入窗口输入密码后，选择要导入的文件，单击“导入”，即可开始导入文件。在完成了文件的导入后，在工作区的输出区域中会显示信息，提示用户导入文件完成。

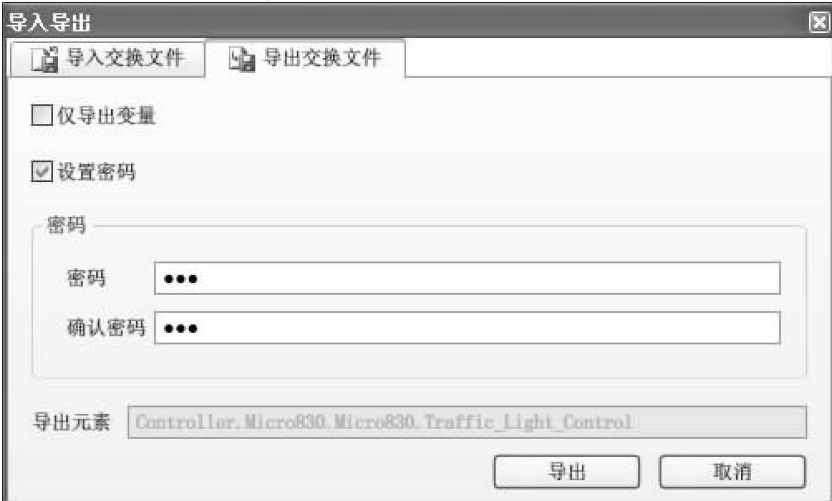


图 5-50　导出程序窗口



图 5-51　导入程序窗口

程序导入完成后，如图 5-52 所示，可以看到新项目中已经包含了导入的程序。



图 5-52　程序导入成功

5.4　程序的调试和下载

完成了程序的编写以后，本节主要介绍程序的下载和调试。

5.4.1　程序的下载和上传

在下载程序之前，首先要建立编程计算机和控制器之间的通信。这里使用 USB 电缆建立通信。把 USB 电缆分别连接到控制器和计算机的 USB 接口上。

打开创建的项目，在项目组织器窗口中双击控制器图标。打开控制器组态窗口，如图 5-53 所示，点击图右上方的“连接”按钮，弹出如图 5-54 所示的连接对话框，找到要连接的控制器型号，这里选择 Micro830 控制器，点击“确定”即可。



图 5-53　连接控制器与计算机

控制器连接计算机以后，组态界面显示如图 5-55 所示，断开连接的按钮为活动状态，同时可以改变控制器的状态，其状态有两种：运行和编程。

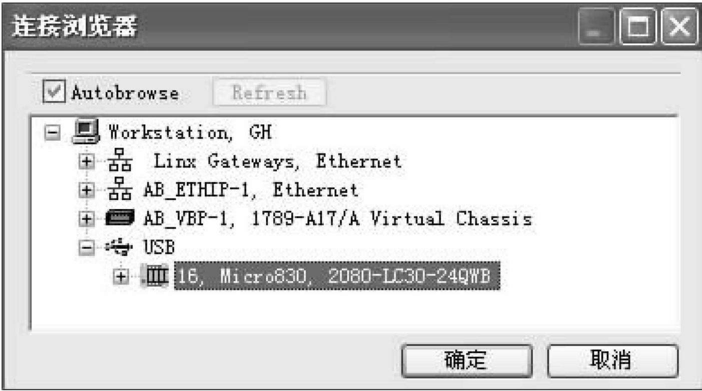


图 5-54　选择要连接的控制器



图 5-55　控制器连接完成后的组态界面显示

连接控制器以后，可以看到在图 5-55 中左边的下载按钮没有处于活动状态，而只有上传按钮处于活动状态。这是因为在下载项目之前要先对项目编译并保存，在项目组织器窗口中，右键单击控制器图标，选择编译（生成）选项，开始编译项目。编译完成后，点击软件左上角的“保存”按钮，保存编译后的项目。



图 5-56　下载项目

保存完成后在项目组织器窗口中右键单击控制器图标，选择下载项目，如图 5-56 所示，也可以在组态窗口中直接选择“下载”按钮，如图 5-57 所示。项目下载过程中，组态界面会显示项目正在下载。下载完成后，会弹出对话框，提示下载完成，是否进入运行状态，用户可以自行选择。



图 5-57　下载按钮

完成项目的下载后，下面讲解项目的上传。把控制器连接到计算机上后，新建一个项目。在项目组织器窗口中右键单击控制器图标，选择上传，如图 5-58 所示。选择上传以后，会弹出连接浏览器对话框，选择目标控制器，这里选择 Micro830 控制器，点击“OK”，会弹出一个对话框，询问是否通过上传来替换当前的项目，选择“是”，项目上传完成后，在软件的工作区输出窗口会弹出信息，提示上传完成。在项目组织器窗口中可以看到上传的项目。



图 5-58　选择上传

5.4.2　程序的调试

在调试程序之前，首先要确定程序处于运行状态，即控制器处于运行模式。然后从主菜单中的调试下拉菜单中选择启动调试，如图 5-59 所示。开始调试后，打开全局变量列表，可以给变量一个强制值，如图 5-60 所示。要强调的是在调试程序之前，一定要确保程序已经被编译通过，并且保存，否则开始调试的选项将不可用。



图 5-59　选择启动调试

此时编程人员可以对项目中的变量强制给值，在没有 I/O 被强制的情况下，可以从程序中看到除了 DO0 和 DO5 为真以外，程序中用到的其他变量都为假，这说明初始状态下南北红灯和东西绿灯是亮的。打开全局变量列表，如图 5-60 所示，可以看到 DO0 和 DO5 的逻辑值和物理值选项均被打钩，与程序中的状态一致。要测试程序，就要强制程序中的一些变

名称	逻辑值	实际值	锁定	数据类型	别名
_IO_EM_DO_00	✓	✓		BOOL	DO0
_IO_EM_DO_01				BOOL	DO1
_IO_EM_DO_02				BOOL	DO2
_IO_EM_DO_03				BOOL	DO3
_IO_EM_DO_04				BOOL	DO4
_IO_EM_DO_05	✓	✓		BOOL	DO5
_IO_EM_DO_06				BOOL	DO6
_IO_EM_DO_07				BOOL	
_IO_EM_DO_08				BOOL	

图 5-60　调试状态的全局变量列表

量，在这里要强制变量之前，必须先把变量锁定，在图 5-60 中，有一列为锁定列，在该列中选定变量就可以对变量进行强制给值。例如：想要强制 DI0，则做如图 5-61 所示的选择。此时程序就会运行，输出的状态变为如图 5-62 所示的状态。与程序中的逻辑一致。

_IO_EM_DO_09	☐	☐	☐	BOOL	
_IO_EM_DI_00	☑	☐	☑	BOOL	DI0
_IO_EM_DI_01	☐	☐	☐	BOOL	DI1

图 5-61　强制 DI0 变量



名称	逻辑值	实际值	锁定	数据类型	别名
_IO_EM_DO_00	☐	☐	☐	BOOL	DO0
_IO_EM_DO_01	☐	☐	☐	BOOL	DO1
_IO_EM_DO_02	☑	☑	☐	BOOL	DO2
_IO_EM_DO_03	☑	☑	☐	BOOL	DO3
_IO_EM_DO_04	☐	☐	☐	BOOL	DO4

图 5-62　输入变量强制后的输出值

想要释放被强制的变量，只要对变量解锁就可以了。用这种方法完成对程序的调试后，就可以停止调试了。在主菜单中的调试菜单的下拉菜单中选择停止调试，就可以停止对程序的调试工作，如图 5-63 所示。

图 5-63　停止程序调试

以上就完成了整个程序的调试。

5.5　小结

CCW 是罗克韦尔的免费编程软件之一。CCW 不仅支持 Micro800 系列控制器的编程，同时还支持罗克韦尔部分触摸屏编程和变频器参数配置。CCW 有着通用、友好的操作界面，灵活易用。它使得编程人员可以在上位机中直接进行梯形图编程，指令的添加用拖拽的方式。添加指令后，会自动弹出变量列表供编程人员选择，可以直接添加已有的变量，也可以新建变量。编程语言上，控制器支持梯形图语言功能块化，可以用梯形图语言编写功能块，

然后在梯形图编程中重复使用，非常方便、灵活。

习　题

1. 试编写 Mciro800 系列控制器的梯形图程序，实现 8 个指示灯的彩灯控制。令其前 20s，按每秒 1 次的速度进行左移位；后 20s，按每秒 1 次的速度进行右移位。循环进行。

2. 如果选用的 Micro800 控制器有模拟量输入/输出功能，请将两个模拟量输入相加，送到模拟量输出。

思考题

1. 测量在数字量输入的由低电平到高电平转换（上升沿）时间和下降沿时间的不同。

2. 利用一个点动式按钮控制电机正转、停止、反转，依次反复。

第 6 章

Micro800 控制器的编程指令

学习目标

- 了解编程器的工作方式
- 了解可编程序控制器编程方式的特点
- 熟练掌握功能块指令、函数指令和操作指令

6.1　Micro800 控制器编程语言

通常 PLC 不采用微机的编程语言，而采用面向控制过程、面向实际问题的自然语言编程。这些编程语言有梯形图、逻辑功能图、布尔代数式等。如罗克韦尔自动化公司所有的 PLC（Micro800、MicroLogix、SLC 500、PLC-5 和 ControlLogix）都支持梯形图（Ladder Diagram，LD）的编程方式。Micro800 控制器支持三种编程方式：梯形图、结构化文本和功能块编程。其最大的特点就是每种编程语言都支持功能块化的编程。下面分别介绍这三种语言。

6.1.1　梯形图

梯形图表达式是在原电器控制系统中常用的接触器、继电器梯形图基础上演变而来的。它沿用了继电器的触点、线圈、串联等术语和图形符号，并增加了一些继电接触控制没有的符号。梯形图形象、直观，对于熟悉继电器方式的人来说，非常容易接受，而不需要学习更深的计算机知识。这是一种最为广泛的编程方式，适用于顺序逻辑控制、离散量控制、定时/计数控制等。

1. 梯形图的格式

梯形图一般由多个不同的梯级（RUNG）组成，每一梯级又由输入及输出指令组成。在一个梯级中，输出指令应出现在梯级的最右边，而输入指令则出现在输出指令的左边，如图 6-1 所示。

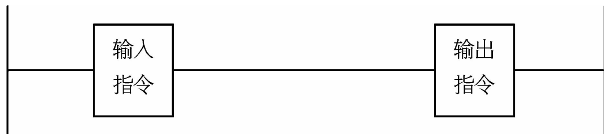


图 6-1　梯形图

当输入指令所表示的梯级条件为真则执行输出指令，否则不执行输出指令。因此，允许在一个梯级中无输入指令——表示梯级条件永远为真；也允许有多个输入指令串并联。串联意味着几个条件之间是“与”的关系，并联则意味着几个条件之间是“或”的关系。输出指令则不允许串联，但允许并联，表示梯级条件为真时，几个输出指令可一并执行。

2. PLC 梯形图的编程特点

1）梯形图中的继电器不是物理继电器，每个继电器实际上是映像寄存器中的一位，因此称为“软继电器”。相应位的状态为“1”，表示该继电器线圈通电，其常开触点闭合，常闭触点断开；相应位的状态为“0”，表示该继电器线圈失电，其常开触点断开，常闭触点闭合。

2）梯形图中流过的电流不是物理电流，而是“概念”电流，是用户程序运算中满足输出执行条件的形象表示方式。“概念”电流只能从左向右流动。

3）梯形图中的继电器接点可在编制用户程序时无限引用，即可常开又可常闭。

4）梯形图中用户逻辑解算结果，马上可为后面用户程序的解算所利用。

5）梯形图输入接点和输出线圈不是物理接点和线圈，用户程序的解算是根据 PLC 内 I/O 映像区每位的状态，而不是解算时现场开关的实际状态。

6）输出线圈只对应输出映像区的相应位，不能用该编程元素直接驱动现场机构，该位

的状态必须通过 I/O 模板上对应的输出单元才能驱动现场执行机构。

7）当 PLC 处于运行状态时，就开始按照梯形图符号排列的先后顺序（从上到下、从左到右）逐一处理，也就是说，PLC 对梯形图是按扫描方式顺序执行程序。

6.1.2 功能块

1. 功能块简介

在 Micro800 控制器中可以用功能块（Function Block Diagram，FBD）编程语言编写一个控制系统中输入和输出之间的控制关系图示。用户也可以使用现有的功能块组合，编辑成需要的用户自定义功能块。

每个功能块都有固定的输入连接点和输出连接点，输入和输出都有固定的数据类型规定。输入点一般在功能块的左边，输出点在右边。

在 FBD 中同样可以使用 LD 编程语言中的元素如：线圈、连接开关按钮、跳转、标签和返回等。与梯形图编程语言所不同的是，在 FBD 编程中所使用的元素放置位置没有太多限制，不像在梯形图中对每个元素有严格规定的位置。且在 FBD 编程语言中同样支持使用功能块操作，如操作指令、函数等大类功能块以及用户自定义的功能块等（只在 Connected Components Workbench™中）。

当使用 FBD 编程时，可以从工具箱拖出功能块元素到编辑框里，并编辑它。图 6-2 所示是一个编程示意图。

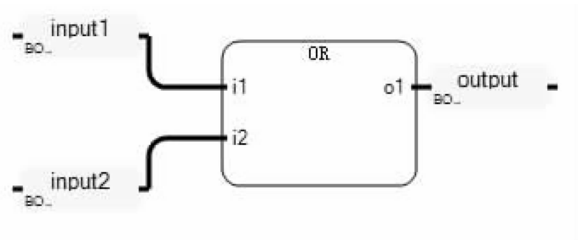


图 6-2　功能块编程示意图

输入和输出变量与 FBD 的输入和输出用连接线连接。信号连接线可以连接如下块的两个逻辑点：一个输入变量和功能块的输入点；功能块的输出和另一功能块的输入点；功能块的输出和输出变量。连接的方向表示连接线带着得到的数据从左边传送到右边。连接线的左右两边必须是相同的数据类型。FBD 多重的右边连接分支也叫做分支结构，可以用于从左边扩展信息至右边。注意数据类型的一致性。

2. 功能块执行顺序

在语言编辑器中，可以显示程序中包含的任意元素的执行顺序（以数字形式）。可以显示 FBD 程序中的以下元素执行顺序：

- 线圈；
- 触点；
- LD 垂直连接；
- 角；
- 返回；
- 跳转；
- 函数；
- 运算符；
- FBD 实例（已声明或未声明）；

●变量（程序中将值分配到的地方）。

注意：当无法确定顺序时，标记显示问号（??）。

要显示执行顺序，请执行以下操作之一：

●按“Ctrl-W”；

●在工具菜单中，选择执行顺序。

在程序执行期间，指令块是 FBD 中的任意元素，网络是链接在一起的一组指令块，指令块的位置是依据其左上角而定的。以下规则适用于 FBD 程序的执行顺序：

●网络从左向右、从上向下执行；

●在执行指令块前，必须解析所有输入。同时解析两个或更多个指令块的输入时，执行决定是根据指令块的位置做出的（从左向右、从上向下）；

●指令块的输出按从左向右、从上向下的顺序以递归方式执行。

3. 调试功能块

调试 FBD 程序时，需要在语言编辑器中监视元素的输出值。这些值使用颜色、数字或文本值加以显示，具体取决于它们的数据类型：

●布尔数据类型的输出值使用颜色进行显示。值为“真”时，默认颜色为红色；值为“假”时，默认颜色为蓝色。输出值的颜色会作为下一输入。输出值不可用时，布尔元素为黑色。

注意：可以在“选项”窗口中自定义用于布尔项的颜色。

SINT、USINT、BYTE、INT、UINT、WORD、DINT、UDINT、DWORD、LINT、ULINT、LWORD、REAL、LREAL、TIME、DATE 和 STRING 数据类型的输出值在元素中显示为数字或文本值。

当数字或文本值的输出值不可用时，在输出标签中会显示问号（??）。值还会显示在对应的变量编辑器实例中。

6.1.3　结构文本

结构文本（Structured Text，ST）类似于 BASIC 编程，利用它可以很方便地建立、编辑和实现复杂的算法，特别是在数据处理、计算存储、决策判断、优化算法等涉及描述多种数据类型的变量应用中非常有效。

1. 结构文本主要语法

ST 程序是一系列 ST 语句。下列规则适用于 ST 程序：

●每个语句以分号（“;”）分隔符结束；

●源代码（例如变量、标识符、常量或语言关键字）中使用的名称用不活动分隔符（例如空格字符）分隔，或者用意义明确的活动分隔符（例如“ > ”分隔符表示“大于”比较）分隔；

●注释（非执行信息）可以放在 ST 程序中的任何位置。注释可以扩展到多行，但是必须以“（ * ”开头，以“ * ）”结尾。

注意：不能在注释中使用注释。

下面是基本 ST 语句类型：

●赋值语句（变量 ： = 表达式;）；

- 函数调用；
- 功能块调用；
- 选择语句（例如 IF、THEN、ELSE、CASE...）；
- 迭代语句（例如 FOR、WHILE、REPEAT...）；
- 控制语句（例如 RETURN、EXIT...）；
- 用于与其他语言链接的特殊语句。

当输入 ST 语法时，下列项目以指定的颜色显示：

- 基本代码（黑色）；
- 关键字（粉色）；
- 数字和文本字符串（灰色）；
- 注释（绿色）。

在活动分隔符、文本和标识符之间使用不活动分隔符可增加 ST 程序的可读性。下面是 ST 不活动分隔符：

- 空格；
- Tab；
- 行结束符（可以放在程序中的任何位置）。

使用不活动分隔符时，需要遵循以下规则：

- 每行编写的语句不能多于一条；
- 使用 Tab 来缩进复杂语句；
- 插入注释以提高行或段落的可读性。

2. 表达式和括号

ST 表达式由运算符及其操作数组成。操作数可以是常量（文本）值、控制变量或另一个表达式（或子表达式）。对于每个单一表达式（将操作数与一个 ST 运算符合并），操作数类型必须匹配。此单一表达式具有与其操作数相同的数据类型，可以用在更复杂的表达式中。

示例：

(boo _ var1 AND boo _ var2)	BOOL 类型
not (boo _ var1)	BOOL 类型
(sin (3.14) + 0.72)	REAL 类型
(t#1s23 + 1.78)	无效表达式

括号用于隔离表达式的子组件，以及对运算的优先级进行明确排序。如果没有为复杂表达式加上括号，则由 ST 运算符之间的默认优先级来隐式确定运算顺序。

示例：

2 + 3 * 6	相当于 2+18=20	乘法运算符具有较高优先级
(2 + 3) * 6	相当于 5*6=30	括号给定了优先级

3. 调用函数和功能块

ST 编程语言可以调用函数。可以在任何表达式中使用函数调用。

函数调用属性见表 6-1。

表 6-1　函数调用属性

属　　性	说　　明
名称	被调用函数的名称以 IEC 61131-3 语言或“C”语言编写
含义	调用结构化文本（ST）、梯形图（LD）或功能块图（FBD）函数或“C”函数，并获取其返回值
语法	: = (, . . .);
操作数	返回值的类型和调用参数必须符合为函数定义的接口
返回值	函数返回的值

当在函数主体中设置返回参数的值时，可以为返回参数赋予与该函数相同的名称：
FunctionName ： = ;
示例
示例 1：IEC 61131-3 函数调用
(* 主 ST 程序 *)
(* 获取一个整型值并将其转换成有限时间值 *)
ana _ timeprog ：= SPlimit (tprog _ cmd)；
appl _ timer ：= ANY _ TO _ TIME（ana _ timeprog * 100）；
(* 被调用的 FBD 函数名为“SPlimit” *)
示例 2：“C”函数调用 – 与 IEC 61131-3 函数调用的语法相同
(* 复杂表达式中使用的函数：min、max、right、mlen 和 left 是标准“C”函数 *)
limited _ value ：= min（16，max（0，input _ value））；
rol _ msg ：= right（message，mlen（message）-1） + left（message，1）；

ST 编程语言调用功能块。可以在任何表达式中使用功能块调用。

功能块调用属性见表 6-2。

表 6-2　功能块调用属性

属　性	说　　明
名称	功能块实例的名称
含义	从标准库中（或从用户定义的库中）调用功能块，访问其返回参数
语法	(* 功能块的调用 *) (, . . .); (* 获取其返回参数 *) : = . ; . . . : = . ;
操作数	参数是与为该功能块指定的参数类型相匹配的表达式
返回值	参见上面的“语法”以获取返回值

当在功能块主体中设置返回参数的值时，可以通过将返回参数的名称与功能块名称相连来分配返回参数：

FunctionBlockName. OutputParaName ： = ；

示例

```
(* 调用功能块的 ST 程序 *)
(* 在变量编辑器中声明块的实例:*)
(* trigb1:块 R _ TRIG -上升沿检测 *)
(* 从 ST 语言激活功能块 *)
trigb1 (b1);
(* 返回参数访问 *)
If (trigb1. Q) Then nb _ edge := nb _ edge + 1; End _ if;
```

6.2　Micro800 控制器的内存组织

Micro800 控制器的内存可以分为两大部分：变量（I/O 变量和中间变量）和程序。下面分别介绍这两部分内容。

6.2.1　变量

Micro800 控制器的变量分为全局变量和本地变量，其中 I/O 变量默认为全局变量。全局变量在项目的任何一个程序或功能块中都可以使用，而本地变量只能在它所在的程序中使用。不同类型的控制器 I/O 变量的类型和个数不同，I/O 变量可以在 CCW 中的全局变量中查看。I/O 变量的名字是固定的，但是可以对 I/O 变量进行别名。除了 I/O 变量以外，为了编程的需要还要建立一些中间变量，变量的类型可以用户自己选择，常用的变量类型见表 6-3。

表 6-3　常用变量类型

变量类型	描　　述	变量类型	描　　述
BOOL	布尔量	LINT	长整型
SINT	单整型	ULINT、LWORD	无符号长整型
USINT、BYTE	无符号单整型	REAL	实型
INT、WORD	整型	LREAL	长实型
UINT	无符号整型	TIME	时间
DINT、DWORD	双整型	DATE	日期
UDINT	无符号双整型	STRING	字符串

在项目组织器中的变量类型中，还可以建立新的变量类型，用来在变量编辑器中定义数组和字，这样方便定义大量相同类型的变量。变量的命名有如下规则：

1）名称不能超过 128 个字符；
2）首字符必须为字母；
3）后续字符可以为字母、数字或者下划线字符。

6.2.2 程序

控制器的程序分为两部分内容：程序（Program）部分（相当于通常的主程序部分）和功能块部分，这里所说的功能块，除了系统自身的函数和功能块指令以外，主要是指用户根据功能需要，自己用梯形图语言编写的具有一定功能的功能块，可以在程序或者功能块中调用，相当于常用的子程序。每个功能块最多有 20 个输入和 20 个输出。Micro810 控制器最多可以有 2000 条含一个操作数的梯级。

在一个项目中可以有多个程序和多个功能块程序。多个程序可以在一个控制器中同时运行，但执行顺序由编程人员设定，设定程序的执行顺序时，在项目组织器中右键单击程序图标，选择属性，打开程序属性对话框，如图 6-3 所示，在 Order 后面写下要执行顺序，1 为第一个执行，2 为第二个执行，例如：一个项目中有 8 个程序，可以把第 8 个程序设定为第一个执行，其他程序会在原来执行的顺序上，依次后推。原来排在第一个执行的程序将自动变为第二个执行。



图 6-3 更改程序执行顺序

6.3 Micro800 控制器的指令系统

罗克韦尔自动化的 PLC 编程指令非常丰富，不同系列 PLC 所支持的指令稍有差异，但基本指令都是大家所共有的。对于编程指令的理解程度，将直接关系到工作的效率。可以这样认为，对编程指令的理解，直接决定了对 PLC 的掌握程度。下面将详细介绍它的指令类型。

6.3.1 梯形图常用指令

编辑梯形图程序时，可以从工具箱拖拽需要的指令符号到编辑窗口中使用。可以添加以下梯形图指令元素：

1. 梯级（Rungs）

梯级是梯形图的组成元素，它表示着一组电子元件激活线圈（输出）。梯级在梯形图中可以有标签，以确定它们在梯形图中的位置。标签和跳转指令（Jumps）配合使用，控制梯形图的执行。梯级示意如图 6-4 所示。

点击编辑框的最左侧，输入该梯级的标签，即对该梯级定义了标签。

2. 线圈（Coils）

线圈（输出）也是梯形图的重要组成元件，它代表着输出或者内部变量。一个线圈代表着一个动作。它的左边必须有布尔元件或者一个指令块的布尔输出。线圈又分为以下几种类型：

直接输出（Direct Coil）

直接输出元件如图 6-5 所示。

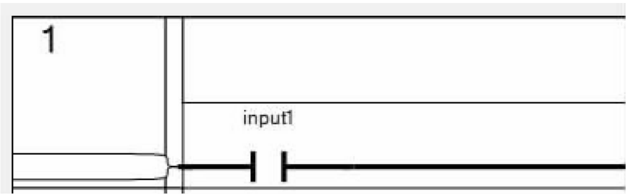


图 6-4　梯形图梯级示意

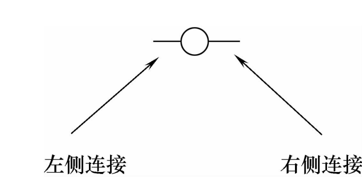


图 6-5　直接输出元件

左连接件的状态直接传送到右连接件上，右连接件必须连接到垂直电源轨上，除非是平行线圈，因为在平行线圈中只有上层线圈必须连接到垂直电源轨上，线圈连接示意如图 6-6 所示。

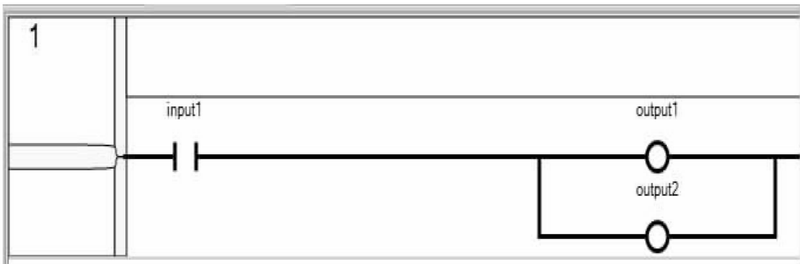


图 6-6　线圈连接示意

反向输出（Reverse Coil）

间接输出元件如图 6-7 所示。

左连接件的反状态被传送到右连接件上，同样，右连接件必须连接到垂直电源轨上，除非是平行线圈。

上升沿（正沿）输出（Pulse Rising Edge Coil）

上升沿（正沿）输出如图 6-8 所示。

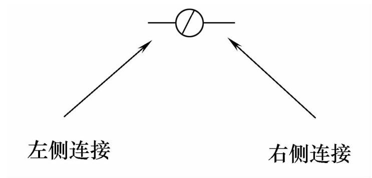


图 6-7　间接输出元件

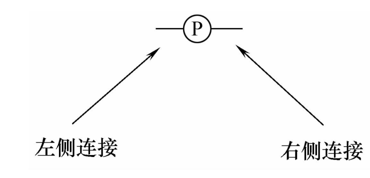


图 6-8　上升沿（正沿）输出

当左连接件的布尔状态由假变为真时，右连接件输出变量将被置 1（即为真），其他情况下输出变量将被重置为 0（即为假）。

下降沿（负沿）输出（Pulse Falling Edge Coil）

下降沿（负沿）输出如图 6-9 所示。

当左连接件的布尔状态由真变为假时，右连接件输出变量将被置 1（即为真），其他情况下输出变量将被重置为 0（即为假）。

置位输出（Set Coil）

置位输出如图 6-10 所示。

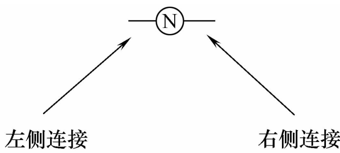


图 6-9　下降沿（负沿）输出

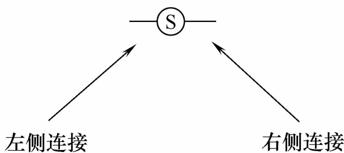


图 6-10　置位输出

当左连接件的布尔状态变为“真”时，输出变量将被置“真”。该输出变量将一直保持该状态直到复位输出（Reset Coil）发出复位命令，置位复位梯形图如图 6-11 所示。

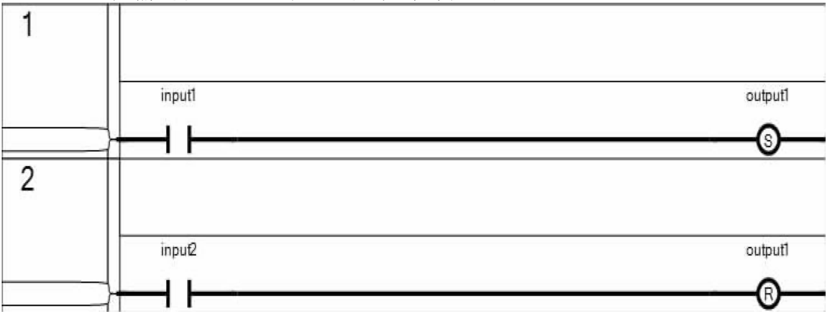


图 6-11　置位复位梯形图

复位输出（Reset Coil）

复位输出如图 6-12 所示。

当左连接件的布尔状态变为“真”时，输出变量将被置“假”。该输出变量将一直保持该状态直到置位输出（Set Coil）发出置位命令。

3. 接触器（Contacts）

接触器在梯形图中代表一个输入的值或是一个内部变量，通常相当于一个开关或按钮的作用。有以下几种连接类型：

直接连接（Direct Contact）

直接连接如图 6-13 所示。

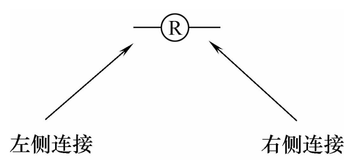


图 6-12　复位输出

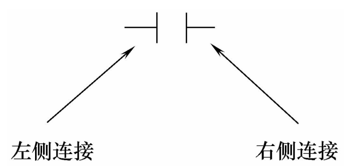


图 6-13　直接连接

左连接件的输出状态和该连接件（开关）的状态取逻辑与，即为右连接件的状态值。

反向连接（Reverse Contact）

反向连接如图 6-14 所示。

左连接件的输出状态和该连接件（开关）的状态的布尔反状态取逻辑与，即为右连接件的状态值。

上升沿（正沿）连接（Pulse Rising Edge Contact）

上升沿（正沿）连接如图 6-15 所示。

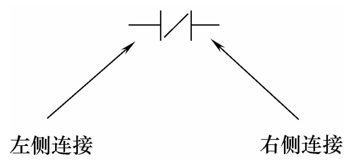


图 6-14　反向连接

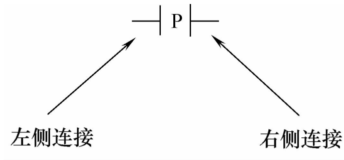


图 6-15　上升沿（正沿）连接

当左连接件的状态为真时，如果该上升沿连接代表的变量状态由假变为真，那么右连接件的状态将会被置“真”，这个状态在其他条件下将会被复位为“假”。

下降沿连接（Pulse Falling Edge Contact）

下降沿连接如图 6-16 所示。

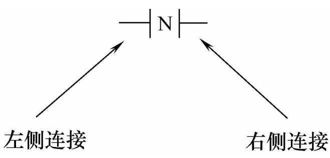


图 6-16　下降沿连接

当左连接件的状态为真时，如果该下降沿连接代表的变量状态由真变为假，那么右连接件的状态将会被置“真”，这个状态在其他条件下将会被复位为“假”。

在现场逻辑控制中，需要对一些操作动作实施互锁来确保执行动作的可靠性。对于几个互锁执行的操作动作，采用锁存解锁指令对其控制是最有效和可靠的，即用如图 6-17 所示的梯级逻辑来确保互锁。

此例中有 4 个互锁的控制，每当满足其中之一的控制条件，便锁存自己的控制，解锁其他控制，不管其他控制当前的状态如何，这样可以确保只有一个控制在执行。这是一种十分可靠的做法，其明了清晰的表达，让读程序的人很容易理解。

4. 指令块（Instruction Blocks）

块（Block）元素指的是指令块，也可以是位操作指令块、函数指令块或者是功能块指令块。在梯形图编辑中，可以添加指令块到布尔梯级中。加到梯级后可以随时用指令块选择

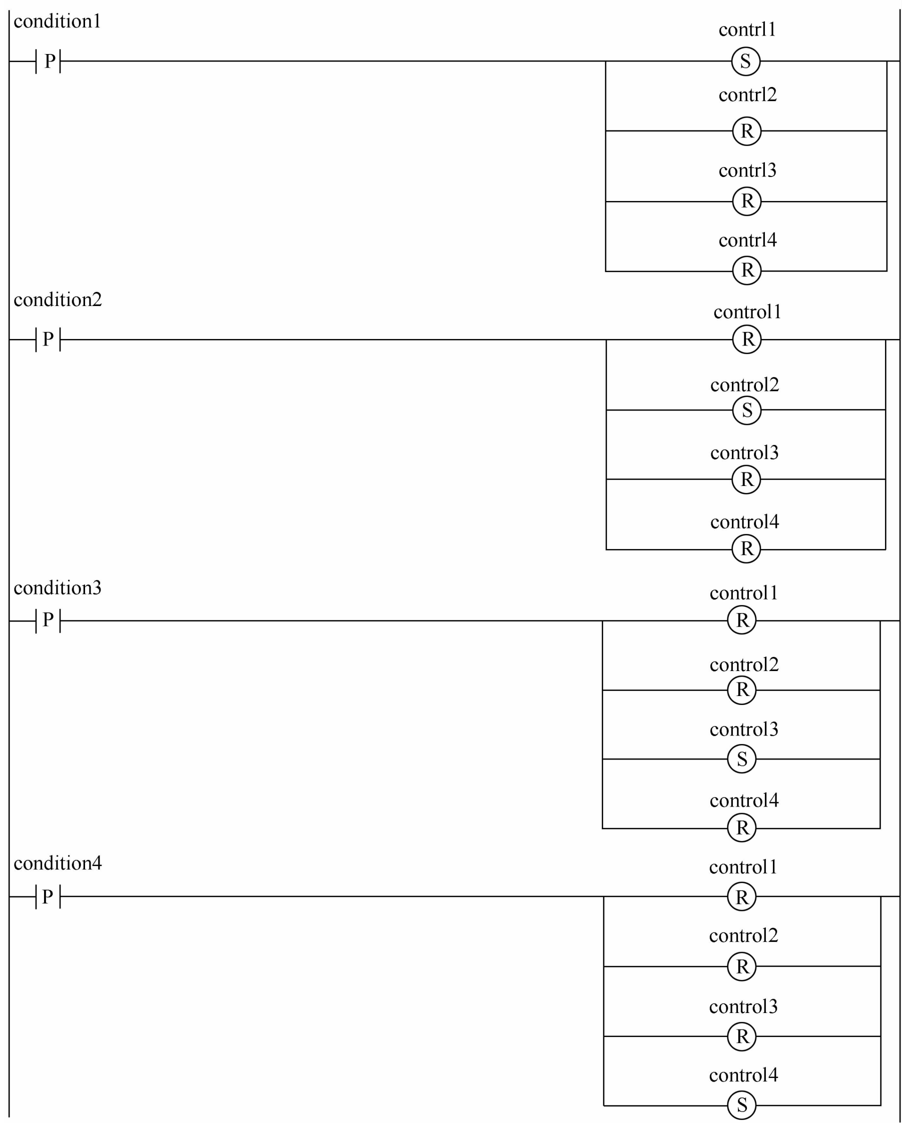


图 6-17　互锁指令梯级逻辑

器设置指令块的类型，随后相关参数将会自动陈列出来。

在使用指令块时请牢记以下两点：

1）当一个指令块添加到梯形图中后，EN 和 ENO 参数将会添加到某些指令块的接口列表中。

2）当指令块是单布尔变量输入、单布尔变量输出或是无布尔变量输入、无布尔变量输出时，可以强制 EN 和 ENO 参数。可以在梯形图操作中激活允许 EN 和 ENO 参数（Enable

EN/ENO）。

从工具箱中拖出块元素放到梯形图的梯级中后，指令块选择器将会陈列出来，为了缩小指令块选择范围，可以使用分类或者过滤指令块列表，或者使用快捷键。

EN 输入

一些指令块的第一输入不是布尔数据类型，由于第一输入总是连接到梯级上的，所以在这种情况下另一种叫 EN 的输入会自动添加到第一输入的位置。仅当 EN 输入为真时，指令块才执行。下面举一个比较指令块的例子，如图 6-18 所示。

ENO 输出

由于第一输出另一端总是连接到梯级上，所以对于第一输出不是布尔型输出的指令块，另一端被称为 ENO 的输出自动添加到了第一输出的位置。ENO 输出的状态总是与该指令块的第一输入的状态一致。下面举一个“平均”指令块的例子，如图 6-19 所示。

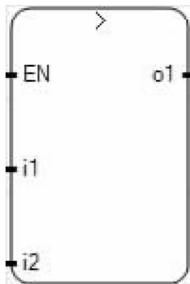


图 6-18　比较指令块

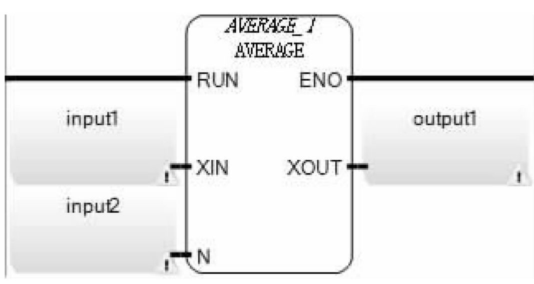


图 6-19　“平均”指令块

EN 和 ENO 参数

在一些情况下，EN 和 ENO 参数都需要。如在数学运算操作指令块中，加法指令块如图 6-20 所示。

功能块使能（Enable）参数

在指令块都需要执行的情况下，需要添加使能参数，例如在“SUS”指令块中，如图 6-21 所示。

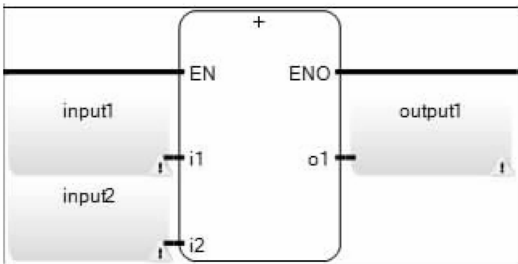


图 6-20　加法指令块

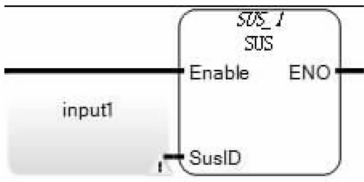


图 6-21　“SUS”指令块

返回（Returns）

当一段梯形图结束时，可以使用返回元件作为输出。注意，不能再在返回元件的右边连接元件。当左边的元件状态为布尔“真”时，梯形图将不执行返回元件之后的指令。当该梯形图为一个函数时，它的名字将被设置为一个输出线圈以设置一个返回值（返回给调用

函数使用)。下面给出一个带返回元件的例子，如图 6-22 所示。

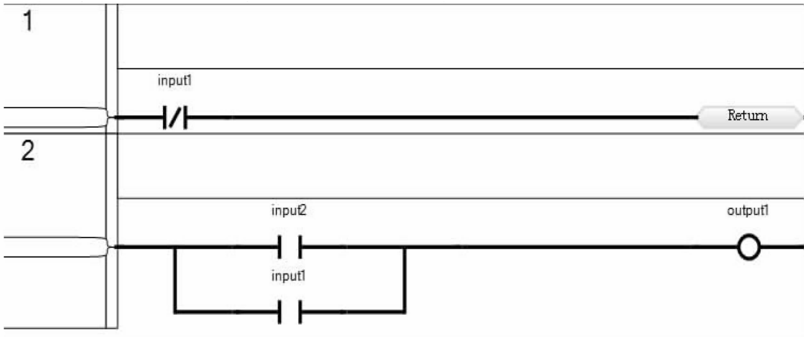


图 6-22　带返回元件的梯形图

5. 跳转（Jumps）

条件和非条件跳转控制着梯形图程序的执行。注意，不能在跳转元件的右边再添加连接件，但可以在其左边添加一些连接件。当跳转元件左边的连接件的布尔状态为“真”时，跳转执行，程序跳转至所需标签处。

6. 分支（Branches）

分支元件能产生一个替代梯级。可以使用分支元件在原来梯级基础上添加一个平行的分支梯级。

6.3.2　功能块指令

功能块指令是 Micro800 控制器编程中的重要指令，它包含了实际应用中的大多数编程功能。功能块指令种类见表 6-4。

表 6-4　功能块指令种类

种　　类	描　　述
报警（Alarms）	超过限制值时报警
布尔运算（Boolean operations）	对信号上升下降沿以及设置或重置操作
通信（Communications）	部件间的通信操作
计时器（Time）	计时
计数器（Counter）	计数
数据操作（Data manipulation）	取平均，最大/最小值
输入/输出（Input/Output）	控制器与模块之间的输入/输出操作
中断（Interrupt）	管理中断
过程控制（Process control）	PID 操作以及堆栈
程序控制（Program control）	主要是延迟指令功能块

1. 报警（Alarms）

功能块指令报警类指令只有限位报警一种，其详细功能说明如下：

限位报警（LIM _ ALRM）

限位报警功能块如图 6-23 所示。

该功能块用高限位和低限位限制一个实数变量。限位报警使用的高限位和低限位是 EPS 参数的一半。其参数列表见表 6-5。

下面简单介绍一下限位报警功能块的用法。限位报警的主要作用就是限制输入，当输入超过或者低于预置的限位安全值时，输出报警信号。在本功能块中 X 接的是实际要限制的输入，其他各参数的意义我们可以参照上表。当 X 的值达到高限位值 H 时，功能块将输出 QH 和 Q，即高位报警和报警，而要解除该报警，需要输入的值小于高限位的滞后值（H-EPS），这样就拓宽了报警的范围，使输入值能较快地回到一个比较安全的范围值内，起到保护机器的作用。对于低位报警，功能块的工作方式很类似。当输入低于低限位值 L 时，功能块输出低位报警（QL）和报警（Q），而要解除报警则需输入回到低限位的滞后值（L + EPS）。可见报警（Q）的输出是综合了高位报警和低位报警。使用时我们可以留意该输出。

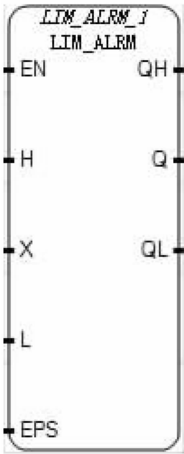


图 6-23　限位报警功能块

表 6-5　限位报警功能块参数列表

参数	参数类型	数据类型	描述
EN	Input	BOOL	功能块使能。为真时，执行功能块；为假时，不执行功能块
H	Input	REAL	高限位值
X	Input	REAL	输入：任意实数
L	Input	REAL	低限位值
EPS	Input	REAL	滞后值（须大于零）
QH	Output	BOOL	高位报警：如果 X 大于高限位值 H 时为真
Q	Output	BOOL	报警：如果 X 超过限位值时为真
QL	Output	BOOL	低位报警：如果 X 小于低限位值 L 时为真

该功能块时序图如图 6-24 所示。

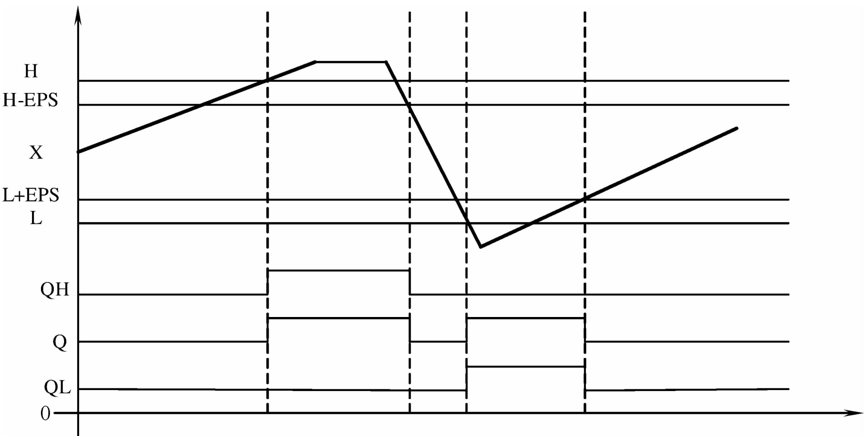


图 6-24　限位报警功能块时序图

2. 布尔操作（Boolean Operations）

布尔操作类功能块主要有以下 4 种，用途见表 6-6。

表 6-6　布尔操作功能块用途

功　能　块	描　　述	功　能　块	描　　述
F _ TRIG（下降沿触发）	下降沿侦测，下降沿时为真	R _ TRIG（上升沿触发）	上升沿侦测，上升沿时为真
RS（重置）	重置优先	SR（设置）	设置优先

下面详细说明下降沿触发以及重置功能块的使用。



图 6-25　下降沿触发功能块

（1）下降沿触发（F _ TRIG）

下降沿触发功能块如图 6-25 所示。

该功能块用于检测布尔变量的下降沿，其参数列表见表 6-7。

表 6-7　下降沿触发功能块参数列表

参数	参数类型	数据类型	描　　述
CLK	Input	BOOL	任意布尔变量
Q	Output	BOOL	当 CLK 从真变为假时，为真；其他情况为假

（2）重置（RS）

重置功能块如图 6-26 所示。

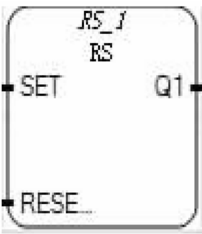


图 6-26　重置功能块

重置优先，其参数列表见表 6-8。

表 6-8　重置功能块参数列表

参数	参数类型	数据类型	描　　述
SET	Input	BOOL	如果为真，则置 Q1 为真
RESET1	Input	BOOL	如果为真，则置 Q1 为假（优先）
Q1	Output	BOOL	存储的布尔状态

重置功能块示例见表 6-9。

表 6-9　重置功能块示例表

SET	RESET1	Q1	Result Q1	SET	RESET1	Q1	Result Q1
0	0	0	0	1	0	0	1
0	0	1	1	1	0	1	1
0	1	0	0	1	1	0	0
0	1	1	0	1	1	1	0

3. 通信（Communications）

通信类功能块主要负责与外部设备通信，以及自身的各部件之间的联系。该类功能块的主要指令见表 6-10。

表 6-10　通信类功能块的主要指令

功　能　块	描　　述
ABL（测试缓冲区数据列）	统计缓冲区中的字符个数（直到并且包括结束字符）
ACB（缓冲区字符数）	统计缓冲区中的总字符个数（不包括终止字符）
ACL（ASCII 清除缓存寄存器）	清除接收，传输缓冲区内容
AHL（ASCII 握手数据列）	设置或重置 RS-232 请求发送（RTS）握手信号控制字
ARD（ASCII 字符读）	从输入缓冲区中读取字符并把它们放到某个字符串中
ARL（ASCII 数据列读）	从输入缓冲区中读取一行字符并把它们放到某个字符串中，包括终止字符
AWA（ASCII 带附加字符写）	写一个带用户配置字符的字符串到外部设备中
AWT（ASCII 字符写出）	从源字符串中写一个字符到外部设备中
MSG _ MODBUS（网络通信协议信息传输）	发送 Modbus 信息

下面主要介绍 ABL，ACL，AHL，ARD，AWA，MSG _ MODBUS 这几种指令：

（1）测试缓冲区数据列（ABL　ASCII Test For Line）

测试缓冲区数据列功能块如图 6-27 所示。



图 6-27　测试缓冲区数据列功能块

测试缓冲区数据列功能块指令可以用于统计在输入缓冲区里的字符个数（一直到并且包括结束字符）。其参数列表见表 6-11。

表 6-11　测试缓冲区数据列功能块参数列表

参　数	参数类型	数据类型	描　述
IN	Input	BOOL	如果是上升沿（IN 由假变真），执行统计
ABLInput	Input	ABLACB（见 ABLACB 数据类型）	将要执行统计的通道
Q	Output	BOOL	假——统计指令不执行；真——统计指令已执行
Characters	Output	UINT	字符的个数
Error	Output	BOOL	假——无错误；真——检测到一个错误
ErrorID	Output	UINT	见 ABL 错误代码

ABLACB 数据类型见表 6-12。

表 6-12　ABLACB 数据类型

参　数	数据类型	描　述
Channel	UINT	串行通道号： 2 代表本地的串行通道口 5 ~ 9 代表安装在插槽 1 ~ 5 的嵌入式模块串行通道口：5 表示在插槽 1；6 表示在插槽 2；7 表示在插槽 3；8 表示在插槽 4；9 表示在插槽 5
TriggerType	USINT（无符号短整型）	代表以下情况中的一种：0：Msg 触发一次（当 IN 从假变为真）；1：Msg 持续触发，即 IN 一直为真；其他值：保留
Cancel	BOOL	当该输入被置为真时，统计功能块指令不执行

ABL 错误代码见表 6-13。

表 6-13　ABL 错误代码

错误代码	描述
0x02	由于数据模式离线，操作无法完成
0x03	由于准备传输信号（Clear-to-Send）丢失，导致传送无法完成
0x04	由于通信通道被设置为系统模式，导致 ASCII 码接收无法完成
0x05	当尝试完成一个 ASCII 码传送时，检测到系统模式（DF1）通信
0x06	检测到不合理参数
0x07	由于通过通道配置对话框停止了通道配置，导致不能完成 ASCII 码的发送或接收
0x08	由于一个 ASCII 码传送正在执行，导致不能完成 ASCII 码写入
0x09	现行通道配置不支持 ASCII 码通信请求
0x0a	取消（Cancel）操作被设置，所以停止执行指令，没有要求动作
0x0b	要求的字符串长度无效或者是一个负数，或者大于 82 或 0。功能块 ARD 和 ARL 中也一样
0x0c	源字符串的长度无效或者是一个负数或者大于 82 或 0。对于 AWA 和 AWT 指令也一样
0x0d	在控制块中的要求的数是一个负数或是一个大于存储于源字符串中字符串长度的数。对于 AWA 和 AWT 指令也一样
0x0e	ACL 功能块被停止
0x0f	通道配置改变

说明："0x" 前缀表示十六进制数。

(2) ASCII 清除缓存寄存器 (ACL ASCII Clear Buffers)

ASCII 清除缓存寄存器功能块如图 6-28 所示。

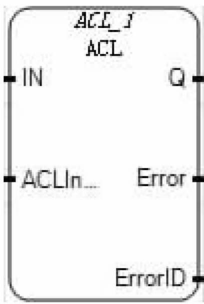


图 6-28　ASCII 清除缓存寄存器功能块

ASCII 清除缓存寄存器功能块指令用于清除缓冲区里的接收和传输的数据，该功能块指令也可以用于移除 ASCII 队列里的指令。其参数见表 6-14。

表 6-14　ASCII 清除缓存寄存器功能块参数

参数	参数类型	数据类型	描　　述
IN	Input	BOOL	如果是上升沿 (IN 由假变真)，执行该功能块
ACLInput	Input	ACL (见 ACL 数据类型)	传送和接收缓冲区的状态
Q	Output	BOOL	假——该功能块不执行；真——该功能块已执行
Error	Output	BOOL	假——无错误；真——检测到一个错误
ErrorID	Output	UINT	见 ABL 错误代码

ACL 数据类型见表 6-15。

表 6-15　ACL 数据类型

参　　数	数据类型	描　　述
Channel	UINT	串行通道号：2 代表本地的串行通道口 5 ~ 9 代表安装在插槽 1 ~ 5 的嵌入式模块串行通道口：5 表示在插槽 1；6 表示在插槽 2；7 表示在插槽 3；8 表示在插槽 4；9 表示在插槽 5
RXBuffer	BOOL	当置为真时，清除接收缓冲区里的内容，并把接收 ASCII 功能块指令 (ARL 和 ARD) 从 ASCII 队列中移除
TXBuffer	BOOL	当置为真时，清除传送缓冲区里的内容，并把传送 ASCII 功能块指令 (AWA 和 AWT) 从 ASCII 队列中移除

(3) ASCII 握手数据列 (AHL ASCII Handshake Lines)

ASCII 握手数据列功能块如图 6-29 所示。

ASCII 握手数据列功能块可以用于设置或重置 RS-232 请求发送 (Request to Send RTS) 握手信号控制行。其参数见表 6-16。

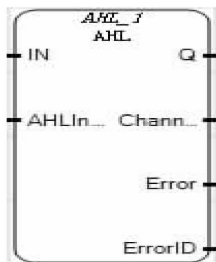


图 6-29　ASCII 握手数据列功能块

表 6-16　ASCII 握手数据列功能块参数

参　　数	参数类型	数据类型	描　　述
IN	Input	BOOL	如果是上升沿（IN 由假变真），执行该功能块
AHLInput	Input	AHL（见 AHLI 数据类型）	设置或重置当前模式的 RTS 控制字
Q	Output	BOOL	假——该功能块不执行；真——该功能块已执行
ChannelSts	Output	WORD（见 AHL ChannelSts 数据类型）	显示当前通道规定的握手行的状态（0000 ~ 001F）
Error	Output	BOOL	假——无错误；真——检测到一个错误
ErrorID	Output	UINT	见 ABL 错误代码

AHLI 数据类型见表 6-17。

表 6-17　AHLI 数据类型

参　　数	数据类型	描　　述
Channel	UINT	串行通道号：2 代表本地的串行通道口 5 ~ 9 代表安装在插槽 1 ~ 5 的嵌入式模块串行通道口：5 表示在插槽 1；6 表示在插槽 2；7 表示在插槽 3；8 表示在插槽 4；9 表示在插槽 5
ClrRts	BOOL	用于重置 RTS 控制字
SetRts	BOOL	用于设置 RTS 控制字
Cancel	BOOL	当输入为真时，该功能块不执行

AHL ChannelSts 数据类型见表 6-18。

表 6-18　AHL ChannelSts 数据类型

参　　数	数据类型	描　　述
DTRstatus	UINT	用于 DTR 信号（保留）
DCDstatus	UINT	用于 DCD 信号（控制字的第 3 位），1 表示激活
DSRstatus	UINT	用于 DSR 信号（保留）
RTSstatus	UINT	用于 RTS 信号（控制字的第 1 位），1 表示激活
CTSstatus	UINT	用于 CTS 信号（控制字的第 0 位），1 表示激活

（4）ASCII 字符读（ARD ASCII Read）

ASCII 字符读功能块如图 6-30 所示。



图 6-30　ASCII 字符读功能块

ASCII 字符读功能块用于从缓冲区中读取字符，并把字符存入一个字符串中。其参数见表 6-19。

表 6-19　ASCII 字符读功能块参数

参数	参数类型	数据类型	描　　述
IN	Input	BOOL	如果是上升沿（IN 由假变真），执行该功能块
ARDInput	Input	ARDARL（见 ARDARL 数据类型）	从缓冲区中读取字符，最多 82 个
Done	Output	BOOL	假——该功能块不执行；真——该功能块已执行
Destination	Output	ASCIILOC	存储字符的字符串位置
NumChar	Output	UINT	字符个数
Error	Output	BOOL	假——无错误；真——检测到一个错误
ErrorID	Output	UINT	见 ABL 错误代码

ARDARL 数据类型见表 6-20。

表 6-20　ARDARL 数据类型

参数	数据类型	描述
Channel	UINT	串行通道号：2 代表本地的串行通道口 5～9 代表安装在插槽 1～5 的嵌入式模块串行通道口：5 表示在插槽 1；6 表示在插槽 2；7 表示在插槽 3；8 表示在插槽 4；9 表示在插槽 5
Length	UINT	希望从缓冲区里读取的字符个数（最多 82 个）
Cancel	BOOL	当输入为真时，该功能块不执行，如果正在执行，则操作停止

（5）ASCII 带附加字符写（AWA ASCII Write Append）

ASCII 带附加字符的写出功能块如图 6-31 所示。

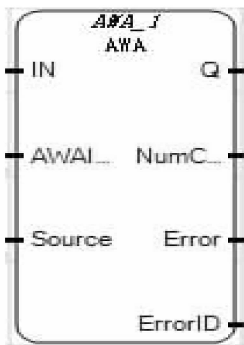


图 6-31　ASCII 带附加字符的写出功能块

写出功能块用于从源字符串向外部设备写入字符。且该指令附加您在设置对话框里设置的两个字符。该功能块的参数列表见表 6-21。

表 6-21　ASCII 带附加字符的写出功能块参数列表

参　数	参数类型	数据类型	描　述
IN	Input	BOOL	如果是上升沿（IN 由假变真），执行功能块
AWAInput	Input	AWAAWT（见 AWAAWT 数据类型）	将要操作的通道和长度
Source	Input	ASCIILOC	源字符串，字符阵列
Q	Output	BOOL	假——功能块不执行；真——功能块已执行
NumChar	Output	UINT	字符个数
Error	Output	BOOL	假——无错误；真——检测到一个错误
ErrorID	Output	UINT	见 ABL 错误代码

AWAAWT 数据类型见表 6-22。

表 6-22　AWAAWT 数据类型

参　数	数据类型	描　述
Channel	UINT	串行通道号：2 代表本地的串行通道口 5～9 代表安装在插槽 1～5 的嵌入式模块串行通道口：5 表示在插槽 1；6 表示在插槽 2；7 表示在插槽 3；8 表示在插槽 4；9 表示在插槽 5
Length	UINT	希望写入缓冲区里的字符个数（最多 82 个）。提示：如果您设置为 0，AWA 将会传送 0 个用户数据字节和两个附加字符到缓冲区
Cancel	BOOL	当输入为真时，该功能块不执行，如果正在执行，则操作停止

（6）网络通信协议信息传输（MSG _ MODBUS）

网络通信协议信息传输功能块如图 6-32 所示。

该功能块用于传送网络通信协议（Modbus）信息，例如读写目标设备的寄存器中的信息。其参数列表见表 6-23。

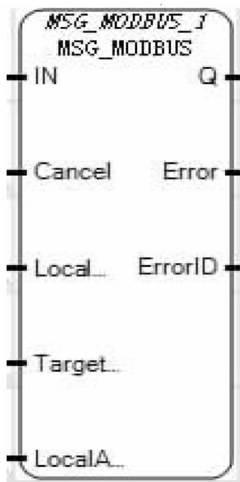


图 6-32　网络通信协议信息传输功能块

表 6-23　网络通信协议信息传输功能块参数列表

参　　数	参数类型	数据类型	描　　述
IN	Input	BOOL	如果是上升沿（IN 从假变为真），执行功能块
Cancel	Input	BOOL	真——取消执行功能块
LocalCfg	Input	MODBUSLOCPARA 见 MODBUSLOCPARA 数据类型	确定结构化输入信息（本地设备）
TargetCfg	Input		确定结构化输入信息（目标设备）
LocalAddr	Input	MODBUSLOCADDR	确定本地存入或写出信息的地址（125 字）MODBUSLOCADDR 数据类型是一个大小为 125 个字的数组，由读取命令用来存储 Modbus 从站返回的数据（1 ~ 125 个字），并由写入命令用来缓冲要发送到 Modbus 从站的数据（1 ~ 125 个字）
Q	Output	BOOL	真——MSG 指令完成；假——指令未完成
Error	Output	BOOL	真——出现错误；假——无错误
ErrorID	Output	UINT	当信息传送错误时，显示错误代码，见 MSG MODBUS 错误代码

MODBUSLOCPARA 数据类型见表 6-24。

表 6-24　MODBUSLOCPARA 数据类型

参　　数	数据类型	描　　述
Channel	UINT	Micro800 PLC 串行端口号：2 代表本地串行端口 5 ~ 9 代表嵌入式串行端口，槽号范围从 1 ~ 5：5 代表槽 1，6 代表槽 2； 7 代表槽 3（Micro830 的 24 点只有 3 个插槽）；8 代表槽 4；9 代表槽 5
TriggerType	USINT	0：Msg 触发一次（IN 从假变为真）； 1：Msg 持续触发，当 IN 为真；其他情况：保留

（续）

参　　数	数据类型	描　　述
Cmd	USINT	MSG 指令的操作命令： 01：读取线圈状态；02：读取输入状态；03：读取保持寄存器； 04：读取输入寄存器；05：写单一线圈；06：写单一寄存器； 15：写多个线圈；16：写多个寄存器
ElementCnt	UINT	读写数据个数的限制： 对于读取线圈或开关量输入最多 2000bits；对于读寄存器最多 125 words； 对于写线圈最多 1968 bits；对于写寄存器最多 123 words

MODBUSTARPARA 数据类型见表 6-25。

表 6-25　MODBUSTARPARA 数据类型

参数	数据类型	描述
Addr	UDINT	目标数据（1 ~ 65536）地址；传送后减 1
Node	USINT	默认从机节点号为 1。节点范围为 0 ~ 247，零是 Modbus 广播节点号，且当 Modbus 处于写命令时有效（如 5，6，15，16）

提示：由于目标数据地址传送后会自动减 1，所以在给 MSG 指令读写地址时，需要在要读写的实际地址基础上加 1 后给到 Addr 上，这样才能使 MSG 指令读写到正确的地址。

MSG _ MODBUS 错误代码见表 6-26。

表 6-26　MSG _ MODBUS 错误代码

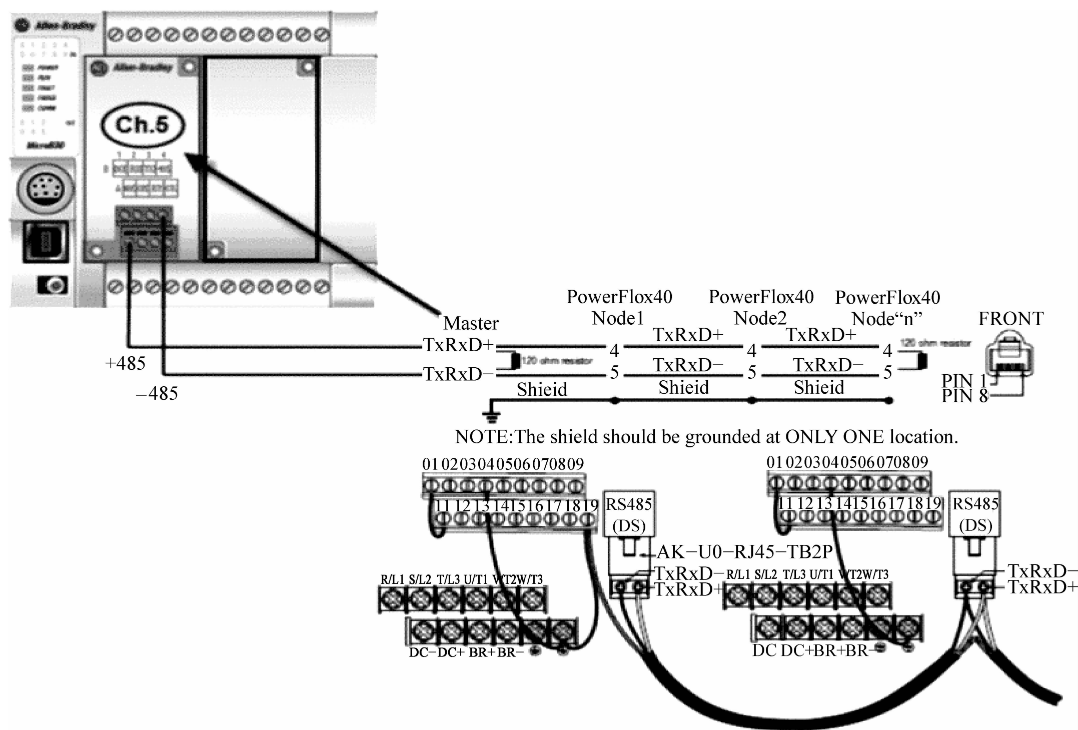
错误代码	描　　述	错误代码	描　　述
3	TriggerType 的类型已经非法改为 2 ~ 255	130	非法数据地址
20	本地通信设备与 MSG 指令不兼容	131	非法数据值
21	本地通道配置参数存在错误	132	从机连接失败
22	目标或本地节点号大于最大允许的节点号	133	响应
33	存在一个损坏的 MSG 文件参数	134	从机忙
54	丢失调制解调设备信息	135	否定响应
55	本地处理器中信息传输超时，链接层超时	136	存储器奇偶校验错误
217	用户取消信息	137	非标准回应
129	非法函数	255	通道被关闭

Modbus 通信组态示例：

本例将演示怎样组态 Modbus 通信，进而使用 MSG _ MODBUS 指令从一个 PowerFlex 4 设备中读取状态数据或是写入控制数据。该例包括以下部分：

1. Micro830 接线

该例使用 Micro830 控制器和一个插到第一个插槽（通道 5，通道定义参见表 6-24）的 SERIALISOL 模块。连接一个独立的 PowerFlex 40，图 6-33 所示的是多终端的接线。更多的接线信息参考用户手册。



AK-U0-RJ45-TB2P is an RJ45 connector with 2 terminal blocks for RS-485 communications

图 6-33　Micro830 控制器和 SERIALISOL 模块接线示意

2. 使用 Modbus 指令读取变频器信息

下面的梯级是使用 MSG _ MODBUS 功能块指令从 PowerFlex 40 设备中读取状态信息，如图 6-34 所示。

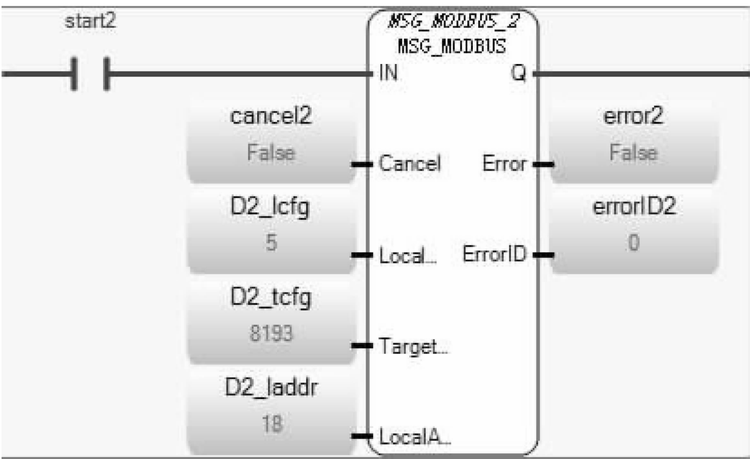


图 6-34　MSG _ MODBUS 功能块指令从 PowerFlex 40 设备中读取状态信息

Modbus 读命令输入/输出变量组态

下面的工作是组态 MSG _ MODBUS 功能块指令，即建立并编辑 MSG 指令的输入/输出变量（注意数据类型的匹配），以读取 PowerFlex 40 设备中的状态信息。组态指令如图 6-35 所示。

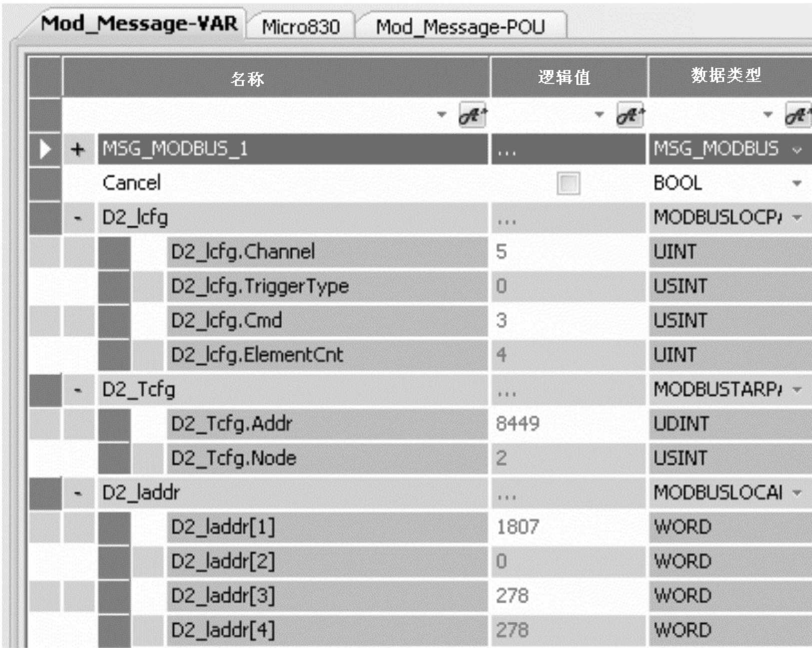


图 6-35　组态 MSG _ MODBUS 功能块指令

图中建立的 MSG _ MODBUS 读变量见表 6-27。

表 6-27　MSG _ MODBUS 读变量

变　　量	值	描　　述
*. Channel	5	通道 5——SERIALISOL 模块所在位置（插槽 1）
*. TriggerType	0	触发假—真的变换，只触发一次
*. Cmd	3	Modbus 功能代码“03”，读保持寄存器
*. ElementCnt	4	读取寄存器的数据长度
*. Addr	8449	PowerFlex 中逻辑状态字地址 +1
*. Node	2	PowerFlex 中的节点地址
* _ [1]	{data}	读取到 MSG 指令上的 PowerFlex 逻辑状态字
* _ [2]	{data}	读取到 MSG 指令上的 PowerFlex 错误代码
* _ [3]	{data}	读取到 MSG 指令上的 PowerFlex 速度命令（速度参考）
* _ [4]	{data}	读取到 MSG 指令上的 PowerFlex 速度反馈（实际速度）

说明：D2 _ laddr [1] 为“1807”（0001 1000 0000 0111）意味着设备准备好（bit 0 为 ON），激活（bit 1 ON），正转命令（bit 2 ON），(3 ON)，与设备的一些数字输入状态一致。

“278” 意味着 27.8Hz。更多信息参考 PowerFlex 用户手册中关于逻辑状态字、位、错误代码描述、命令、实际速度和其他状态代码。

3. 使用 Modbus 指令写信息到变频器

下面是使用 MSG _ MODBUS 功能块指令向 PowerFlex 40 设备写入控制数据，如图 6-36 所示。

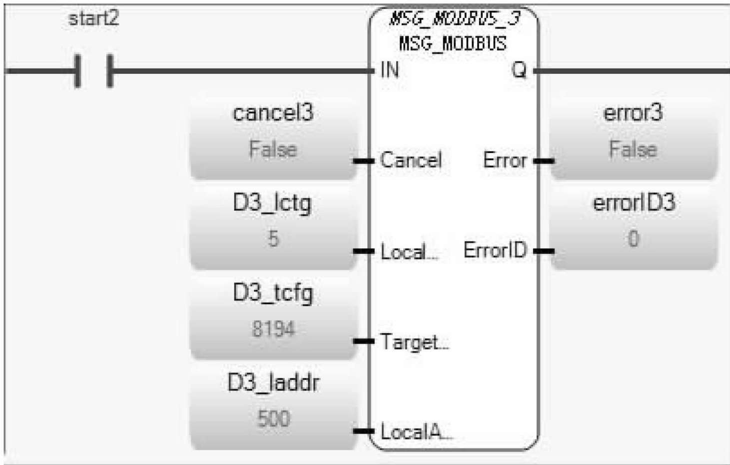


图 6-36　MSG _ MODBUS 功能块指令向 PowerFlex 40 设备写入控制数据

MSG _ MODBUS 写命令输入/输出变量组态

图 6-37 所示是 MSG _ MODBUS 写命令组态，通过对 MSG _ MODBUS _3 指令进行如下的组态，我们可以向 PowerFlex 40 设备中写入控制信息。

Mod_Message-VAR | Micro830 | Mod_Message-POU

名称	逻辑值	数据类型	别名
- MSG_MODBUS_2	...	MSG_MODBUS	
- D3_lcfg	...	MODBUSLOCP/	
D3_lcfg.Channel	5	UINT	
D3_lcfg.TriggerType	0	USINT	
D3_lcfg.Cmd	16	USINT	
D3_lcfg.ElementCnt	2	UINT	
- D3_Tcfg	...	MODBUSTARP/	
D3_Tcfg.Addr	8193	UDINT	
D3_Tcfg.Node	2	USINT	
- D3_laddr	...	MODBUSLOCAI	
D3_laddr[1]	18	WORD	
D3_laddr[2]	278	WORD	

图 6-37　MSG _ MODBUS 写命令组态

MSG _ MODBUS _3 的写变量见表 6-28。

表 6-28　MSG _ MODBUS 写变量

变　　量	值	描　　述
*. Channel	5	通道 5——SERIALISOL 模块所在位置
*. TriggerType	0	触发类型，假—真的变换触发一次
*. Cmd	16	Modbus 功能代码“16”写保持寄存器
*. ElementCnt	2	欲写入变频器的数据长度
*. Addr	8193	PowerFlex 逻辑命令字地址 +1
*. Node	2	PowerFlex 节点地址
*_[1]	{data}	欲写到 PowerFlex 上的逻辑命令字
*_[2]	{data}	欲写到 PowerFlex 上的速度参考字

4. 计数器（Counter）

计数器功能块指令主要用于增减计数，其主要指令见表 6-29。

表 6-29　计数器功能块指令

功　能　块	描　　述	功　能　块	描　　述
CTD(减计数)	减计数	CTUD(给定加减计数)	增减计数
CTU(增计数)	增计数		

下面主要介绍给定加减计数功能块指令：

给定加减计数（CTUD）

给定加减计数功能块如图 6-38 所示。

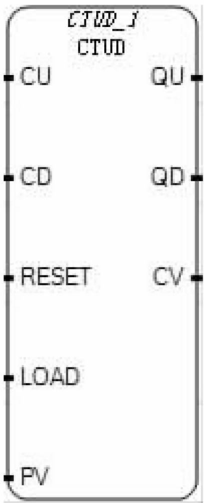


图 6-38　给定加减计数功能块

从 0 开始加计数至给定值，或者从给定值开始减计数至 0。其参数列表见表 6-30。

表 6-30　给定加减计数功能块参数列表

参　数	参数类型	数据类型	描　述
CU	Input	BOOL	加计数(当 CU 是上升沿时,开始计数)
CD	Input	BOOL	减计数(当 CD 是上升沿时,减计数)
RESET	Input	BOOL	重置命令(高级)(RESET 为真时 CV =0 时)
LOAD	Input	BOOL	加载命令(高级)(当 LOAD 为真时 CV = PV)
PV	Input	DINT	程序最大值
QU	Output	BOOL	上限,当 CV≥PV 时为真
QD	Output	BOOL	上限,当 CV≤0 时为真
CV	Output	DINT	计数结果

下面用一个例子讲解计数器的使用方法，程序如图 6-39 所示。

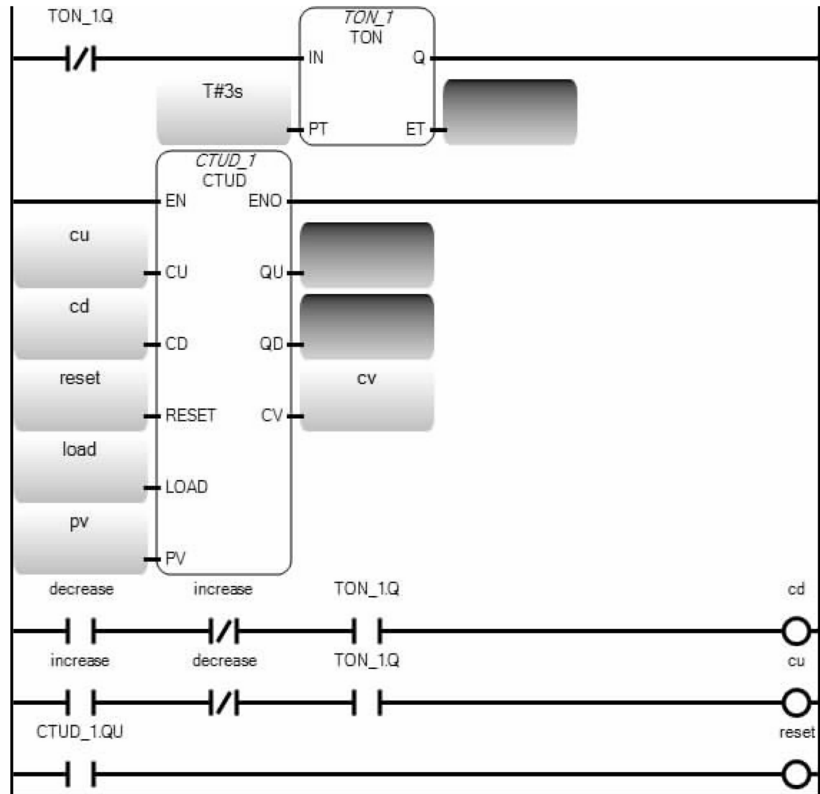


图 6-39　加减计数程序

这个程序要实现的功能是加减计数，梯级一是一个自触发的计时器，TON _ 1. Q 每 3s 输出一个动作脉冲，并复位计时器，重新计时。梯级二使能加减计数器模块。梯级三通过 decrease 位使能减计数，这时当 TON _1. Q 位输出一个脉冲时，PV 值减一。同理，梯级四用来使能加计数。梯级五用来复位加减计数器。这样便实现了加减计数功能。

5. 计时器（Time）

计时器类功能块指令主要有以下 4 种，其指令见表 6-31。

表 6-31　计时器功能块指令

功　能　块	描　　述	功　能　块	描　　述
TOF(延时断增计时)	延时断计时	TONOFF(延时通延时断)	在为真的梯级延时通,在为假的梯级延时断
TON(延时通增计时)	延时通计时	TP(上升沿计时)	脉冲计时

下面详细介绍上述指令：

（1）延时断增计时（TOF）

延时断增计时功能块如图 6-40 所示。



图 6-40　延时断增计时功能块

增大内部计时器至给定值。其参数列表见表 6-32。

表 6-32　延时断增计时功能块参数列表

参　　数	参数类型	数据类型	描　　述
IN	Input	BOOL	下降沿,开始增大内部计时器;上升沿,停止且复位内部计时器
PT	Input	TIME	最大编程时间,见 Time 数据类型
Q	Output	BOOL	真:编程的时间没有消耗完
ET	Output	TIME	已消耗的时间,范围:0 ~ 1193h2m47s294ms。注:如果您在该功能块使用 EN 参数,当 EN 置真时,计时器开始增计时,且一直持续下去(即使 EN 变为假)

该功能块时序图如图 6-41 所示。

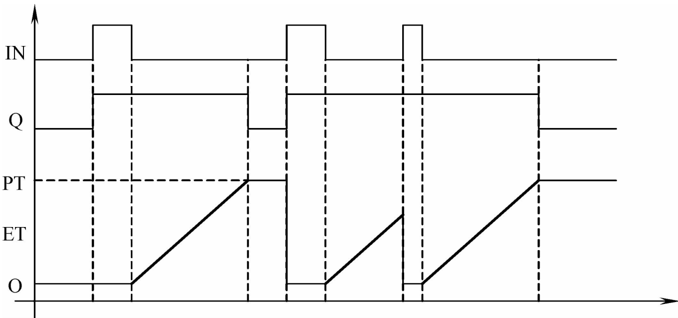


图 6-41　延时断增计时功能块时序图

我们研究一下该时序图。延时断功能块其本质就是输入断开（即下降沿）一段时间（达到计时值）后，功能块输出（即 Q）才从原来的通状态（1 状态）变为断状态（0 状态），即延时断。从图 6-41 中我们可以看出梯级条件 IN 的下降沿才能触发计时器工作，且当计时未达到预置值（PT）时，如果 IN 又有下降沿时，计时器将从新开始计时。参数 ET 表示的是已消耗的时间，即从计时开始到目前为止计时器统计的时间，明显可以看出，ET 的取值范围是（0，PT 的设置值）。输出 Q 的状态由两个条件控制，从时序图中可以看出：当 IN 为上升沿时，Q 开始从 0 变为 1，前提是原来的状态是 0，如果原来的状态是 1，即上次计时没有完成，则如果又碰到 IN 的上升沿，Q 保持原来的 1 的状态；当计时器完成计时时，Q 才回复到 0 状态。所以 Q 由 IN 的状态和计时器完成情况共同控制。

下面用一个例子讲解延时断增计时（TOF）的使用方法。延时断开梯级逻辑如图 6-42 所示。

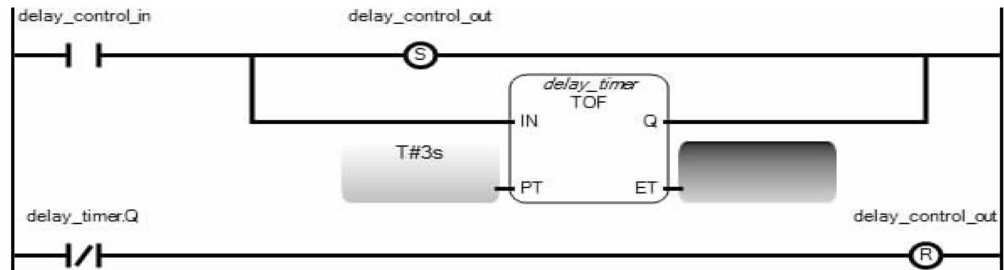


图 6-42　延时断开梯级逻辑

如图 6-42 所示，当 delay _ control _ in 置 1 时，delay _ control _ out 置位，此时 delay _ timer. Q 位保持为 1。当 delay _ control _ in 由 1 变为 0 时，断电延时计时器开始计时，计时 3s 后，delay _ timer. Q 位由 1 变为 0，梯级二导通，delay _ control _ out 复位。由此便实现了断电延时的功能。

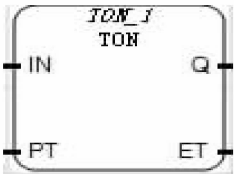


图 6-43　延时通增计时功能块

（2）延时通增计时（TON）

延时通增计时功能块如图 6-43 所示。

增大内部计时器至给定值。其参数列表见表 6-33。

表 6-33　延时通增计时功能块参数列表

参　数	参数类型	数据类型	描　述
IN	Input	BOOL	上升沿，开始增大内部计时器；下降沿，停止且重置内部计时器
PT	Input	TIME	最大编程的时间，见 Time 数据类型
Q	Output	BOOL	真：编程的时间已消耗完
ET	Output	TIME	已消耗的时间，允许值：0 ~ 1193h2m47s294ms。注：如果您在该功能块使用 EN 参数，当 EN 置真时，计时器开始增计时，且一直持续下去（即使 EN 变为假

该功能块时序图如图 6-44 所示。

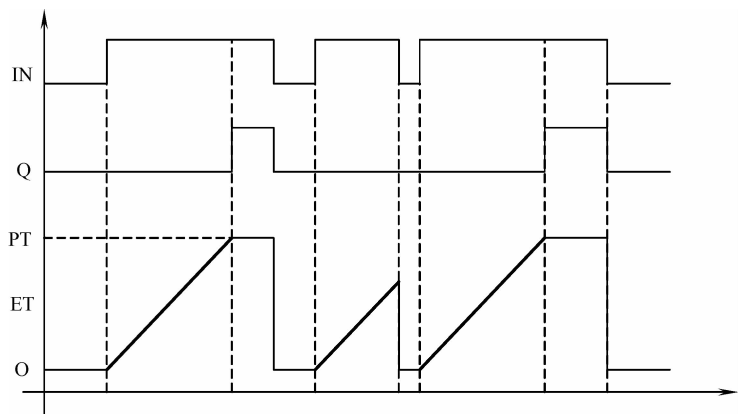


图 6-44　延时通增计时功能块时序图

我们研究一下该时序图。延时通功能块的实质是输入 IN 导通后，输出 Q 延时导通。从图 6-44 中我们可以看出梯级条件 IN 的上升沿触发计时器工作，IN 的下降沿能直接停止计时器计时。参数 ET 表示的是已消耗的时间，即从计时开始到目前为止计时器统计的时间，明显可以看出，ET 的取值范围也是（0，PT 的设置值）。输出 Q 的状态也是由两个条件控制，从时序图中可以看出：当 IN 为上升沿时，计时器开始计时，达到计时时间后 Q 开始从 0 变为 1；直到 IN 变为下降沿时，Q 才跟着变为 0；当计时器未完成计时时，即 IN 的导通时间小于预置的计时时间，Q 将仍然保持原来的 0 状态。

下面用一个例子讲解延时通增计时（TON）的使用方法。通电延时梯级逻辑如图 6-45 所示。

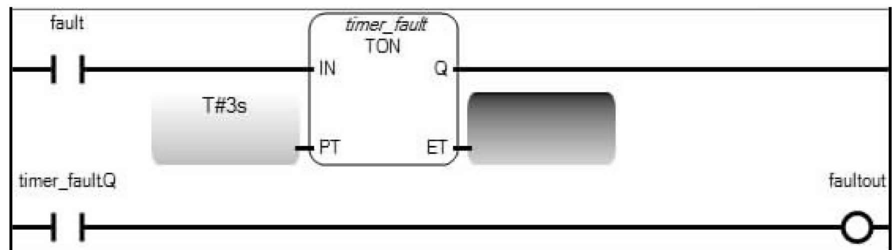


图 6-45　通电延时梯级逻辑

如图 6-45 所示，这个程序在现场常用于检测故障信号，当探测故障发生的信号进来，如果马上动作，可能会引起停机，因为有的故障是需要停机的，假定这个故障信号并不是真正的故障，可能只是一个干扰信号，停机就变得虚惊一场了。所以一般情况下会将这个信号延时一段，确定故障真实存在，再去故障停机。本程序便是使用了延时通增计时（TON）来实现这一功能。

我们将计时器的预定值定义为 3s，那么 TON 的梯级条件 fault 能保持 3s，则故障输出动作的产生将延时 1s 执行。如果这是一个扰动信号，不到 3s 便已经消失，计时器 TON 的梯级条件随之消失，计时器复位，完成位不会置位，故障输出动作不会发生。故障动作延时时间可以根据现场实际情况来确定，挑选一个合适的延时时间即可。

（3） 延时通延时断（TONOFF）

延时通延时断功能块如图 6-46 所示。

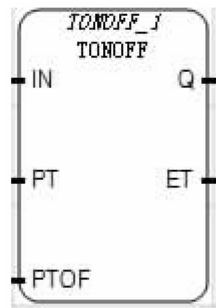


图 6-46　延时通延时断功能块

该功能块用于在输出为真的梯级中延时通，在为假的梯级中延时断开。其参数列表见表 6-34。

表 6-34　延时通延时断功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
IN	Input	BOOL	如果 IN 上升沿，延时通计时器开始。如果程序设定的延时通时间消耗完毕，且 IN 是下降沿（从 1 到 0），延时断计时器开始计时，且重置已用时间（ET）。如果程序延时通时间没有消耗完毕，且处于上升沿，继续开启延时通计时器
PT	Input	TIME	延时通时间设置
PTOF	Input	TIME	延时断时间设置
Q	Output	BOOL	真：程序延时通时间消耗完毕，程序延时断时间没有消耗完毕
ET	Output	TIME	当前消耗时间。允许值：0 ~ 1193h2m47s294ms。如果程序延时通时间消耗完毕且延时断计时器没有开启，消耗时间（ET elapsed time）保持在延时通的时间值（PT） 如果设定的关断延时时间已过，且关断延时计时器未启动，则上升沿再次发生之前，消耗时间（ET）仍为关断延时（PTOF）值 如果延时断的时间消耗完毕，且延时通计时器没有开启，则消耗时间保持与延时断的时间值（PTOF）一致，直到上升沿再次出现为止 注：如果您在该功能块使用 EN 参数，当 EN 为真时，计时器开始增计时，且持续下去（即使 EN 被置为假）

下面通过一个例子讲解延时通延时断（TONOFF）的用法。

延时通延时断梯级逻辑如图 6-47 所示。

如图 6-47 所示，这个例子是某个输出开关的控制要求，当控制发出打开命令后，延时 3s 打开；控制发出关闭命令后，延时 2s 关闭。如果发出打开的命令后不到 3s 接收关闭命令，则不打开；如果发出关闭命令后不到 2s 接收打开命令，则不关闭。

通过 TONOFF 指令，很轻松地实现了这一功能。延时控制开关 in 作为 TONOFF 的梯级条件，开或关的任意情况会触发通电计时或断电计时，从而控制 out 位输出。

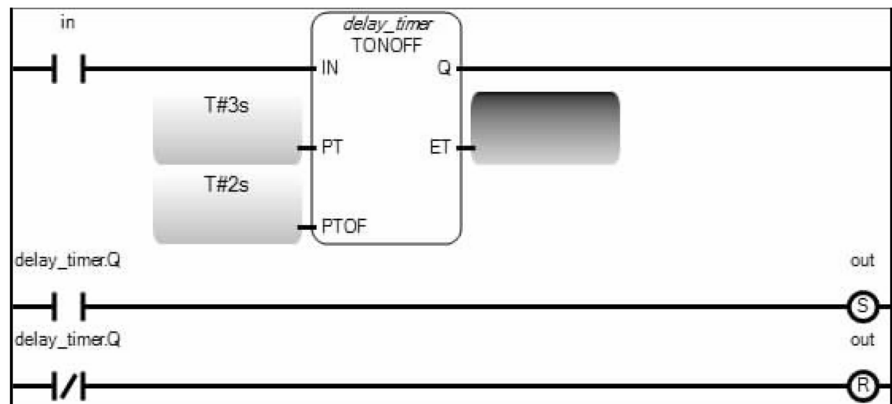


图 6-47　延时通延时断梯级逻辑

（4） 上升沿计时（TP）

上升沿计时功能块如图 6-48 所示。

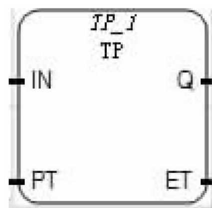


图 6-48　上升沿计时功能块

在上升沿，内部计时器增计时至给定值，若计时时间达到，则重置内部计时器。其参数列表见表 6-35。

表 6-35　上升沿计时功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
IN	Input	BOOL	如果 IN 上升沿,内部计时器开始增计时(如果没有开始增计时) 如果 IN 为假且计时时间到,重置内部计时器。在计时期间任何改变将无效
PT	Input	TIME	最大编程时间
Q	Output	BOOL	真:计时器正在计时
ET	Output	TIME	当前消耗时间。允许值:0 ~ 1193h2m47s294ms 注:如果您在该功能块使用 EN 参数,当 EN 为真时,计时器开始增计时,且持续下去(即使 EN 被置为假)

该功能块时序图如图 6-49 所示。

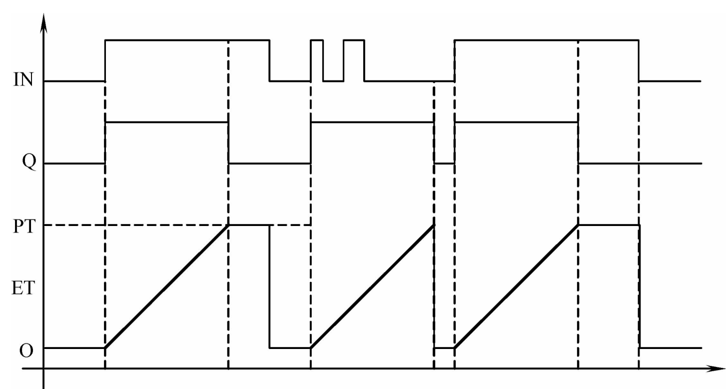


图 6-49　上升沿计时功能块时序图

下面我们研究一下该功能块的时序图。从时序图我们可以看出，上升沿计时功能块与其他功能块明显的不同是其消耗时间（ET）总是与预置值（PT）相等。我们可以看出，输入 IN 的上升沿触发计时器开始计时，当计时器开始工作后，就不受 IN 干扰，直至计时完成。计时器完成计时后才接受 IN 的控制，即计时器的输出值保持住当前的计时值，直至 IN 变为 0 状态时，计时器才回到 0 状态。此外，输出 Q 也与之前的计时器不同，计时器开始计时时，Q 由 0 变为 1，计时结束后，再由 1 变为 0。所以 Q 可以表示计时器是否在计时状态。

6. 数据操作（Data Manipulation）

数据操作类功能块主要有最大值和最小值，其用途描述见表 6-36。

表 6-36　数据操作类功能块用途描述

功　能　块	描　　述
AVERAGE(平均)	取存储数据的平均
MAX(最大值)	比较产生两个输入整数中的最大值
MIN(最小值)	计算两个整数输入中最小的数

下面举例说明该类功能块的参数及应用。

平均（AVERAGE）

平均功能块如图 6-50 所示。

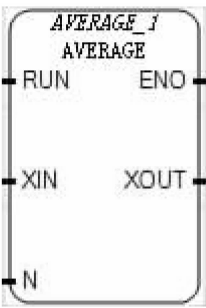


图 6-50　平均功能块

平均功能块用于计算每一循环周期所有已存储值的平均值，并存储该平均值。只有 N 的最后输入值被存储。N 的样本数个数不能超过 128 个。如果 RUN 命令为假（重置模式），

输出值等于输入值。当达到最大的存储个数时，第一个存储的数将被最后一个替代。该功能块的参数列表见表 6-37。

表 6-37　平均功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
RUN	Input	BOOL	真 = 执行、假 = 重置
XIN	Input	REAL	任何实数
N	Input	DINT	用于定义样本个数
XOUT	Output	REAL	输出 XIN 的平均值
ENO	Output	BOOL	使能输出
提示:需要设置或更改 N 的值时,需要把 RUN 置假,然后置回真			

7. 输入/输出（Input/Output）

输入/输出类功能块指令主要用于管理控制器与外设之间的输入和输出数据，其指令用途见表 6-38。

表 6-38　输入/输出类功能块指令用途

功　能　块	描　　述
HSC(高速计数器)	设置要应用到高速计数器上的高和低预设值以及输出源
HSC _ SET _ STS(HSC 状态设置)	手动设置/重置高速计数器状态
IIM(立即输入)	在正常输出扫描之前更新输入
IOM(立即输出)	在正常输出扫描之前更新输出
KEY _ READ(键状态读取)	读取可选 LCD 模块中的键的状态(只限 Micro810™)
MM _ INFO(存储模块信息)	读取存储模块的标题信息
PLUGIN _ INFO(嵌入型模块信息)	获取嵌入型模块信息(存储模块除外)
PLUGIN _ READ(嵌入型模块数据读取)	从嵌入型模块中读取信息
PLUGIN _ RESET(嵌入型模块重置)	重置一个嵌入型模块(硬件重置)
PLUGIN _ WRITE(写嵌入型模块)	向嵌入型模块中写入数据
RTC _ READ(读 RTC)	读取实时时钟(RTC)模块的信息
RTC _ SET(写 RTC)	向实时时钟模块设置实时时钟数据
SYS _ INFO(系统信息)	读取 Micro800™ 系统状态
TRIMPOT _ READ(微调电位器)	从特定的微调电位模块中读取微调电位值
LCD(显示)	显示字符串和数据(只限于 Micro810™)
RHC(读高速时钟的值)	读取高速时钟的值
RPC(读校验和)	读取用户程序校验和

下面将详细介绍上述指令块：

（1）高速计数器（HSC）

高速计数器功能块如图 6-51 所示。

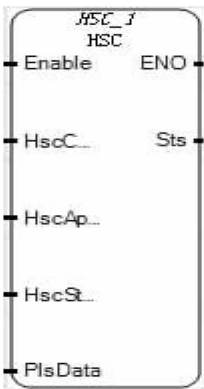


图 6-51　高速计数器功能块

该功能块用于启/停高速计数，刷新高速计数器的状态，重载高速计数器的设置，以及重置高速计数器的累加值。

注意：在 CCW 中高速计数器被分为两个部分，高速计数部分和用户接口部分。这两部分是结合使用的。本小节主要介绍高速计数部分。用户接口部分由一个中断机制驱动（例如中断允许（UIE）、激活（UIF）、屏蔽（UID）或是自动允许中断（AutoStart）），用于在高速计数器到达设定条件时驱动执行指定的用户中断程序。该功能块的参数列表见表 6-39。

表 6-39　高速计数器功能块参数列表

参　　数	参数类型	数据类型	描　　述
HscCmd	Input	USINT	功能块执行、刷新等控制命令，见 HSC 命令参数
HSCAppData	Input	HSCAPP	HSC 应用配置。通常只需配置一次。见 HSC 应用数据结构
HSCStsInfo	Input	HSCSTS	HSC 动态状态。通常在 HSC 执行周期里该状态信息会持续更新，见 HSC 状态信息数据结构
PlsData	Input	PLS	可编程限位开关数据（Programmable Limit Switch，PLS），用于设置 HSC 的附加高低及溢出设定值。见 PLS 数据类型
Sts	Output	UINT	HSC 功能块执行状态，见 HSC 状态值

HSC 命令参数（HscCmd），见表 6-40。

表 6-40　HSC 命令参数

HSC 命令	命令描述
0x00	保留，未使用
0x01	执行 HSC：运行 HSC（如果 HSC 处于空闲模式且梯级使能）；只更新 HSC 状态信息（如果 HSC 处于运行模式，且梯级使能）
0x02	停止 HSC，如果 HSC 处于运行模式，且梯级使能
0x03	上载或设置 HSC 应用数据配置信息（如果梯级使能）
0x04	重置 HSC 累加值（如果梯级使能）

说明："0x"前缀表示十六进制数。

HSCAPP 数据类型（HSCAppData）见表 6-41。

表 6-41　HSCAPP 数据类型

参　　数	数 据 类 型	描　　述
PLSEnable	BOOL	使能或停止可编程限位开关(PLS)
HscID	UINT	要驱动的 HSC 编号,见 HSC ID 定义
HSCMode	UINT	要使用的 HSC 计数模式,见 HSC 模式
Accumulator	DINT	设置计数器的计数初始值
HPSetting	DINT	高预设值
LPSetting	DINT	低预设值
OFSetting	DINT	溢出设置值
UFSetting	DINT	下溢设置值
OutputMask	UDINT	设置输出掩码
HPOutput	UDINT	高预设值的 32 位输出值
LPOutput	UDINT	低预设值的 32 位输出值

说明：OutputMask 指令的作用是屏蔽 HSC 输出的数据中的某几位，以获取期望的数据输出位。例如，对于 24 点的 Micro830，有 9 点本地（控制器自带）输出点用于输出数据，当不需输出第零位的数据时，可以把 OutputMask 中的第零位置 0 即可。这样即使输出数据上的第零位为 1，也不会输出。

HscID、HSCMode、HPSetting、LPSetting、OFSetting、UFSetting 6 个参数必须设置，否则将提示 HSC 配置信息错误。上溢值最大为 +2，147，483，647，下溢值最小为 -2，147，483，647。预设值大小须对应，即高预设值不能比上溢值大，低预设值不能比下溢值小。当 HSC 计数值达到上溢值时，会将计数值置为下溢值继续计数；达到下溢值时，类似。

HSC 应用数据是 HSC 组态数据，它需要在启动 HSC 前组态完毕。在 HSC 计数期间，该数据不能改变，除非需要重载 HSC 组态信息（在 HscCmd 中写 03 命令）。但是，在 HSC 计数期间的 HSC 应用数据改变请求将被忽略。

HSC ID 定义见表 6-42。

表 6-42　HSC ID 定义

位	描　　述
15 ~ 13	HSC 的模式类型:0x00——本地;0x01——扩展式(暂无);0x02——嵌入式
12 ~ 8	模块的插槽 ID:0x00——本地;0x01 ~ 0x1F——扩展式(暂无)模块的 ID 0x01 ~ 0x05——嵌入式模块的 ID
7 ~ 0	模块内部的 HSC ID:0x00 ~ 0x0F——本地;0x00 ~ 0x07——扩展式(暂无);0x00 ~ 0x07——嵌入式 注意:对于初始版本的 Connected Components Workbench 只支持 0x00 ~ 0x05 范围的 ID

使用说明：将表中各位上符合实际要使用的 HSC 的信息数据组合为一个无符号整数，写到 HSCAppData 的 HscID 位置上即可。例如，选择控制器自带的第一个 HSC 接口，即 15

~13 位为 0，表示本地的 I/O；12 ~8 位为 0，表示本地的通道，非扩展或嵌入模块；7 ~0 位为 0，表示选择第 0 个 HSC，这样最终就在定义的 HSCAPP 类型的输入上的 HscID 位置上写入 0 即可。

HSC 模式（HSCMode）见表 6-43 所示。

表 6-43　HSC 模式

模式	功　能	模式	功　能
0	递增计数	5	有“重置”和“保持”控制信号的两输入计数
1	有外部“重置”和“保持”控制信号的递增计数	6	正交计数(编码形式,有 A、B 两相脉冲)
2	双向计数,并带有“外部方向”控制信号	7	有“重置”和“保持”控制信号的正交计数
3	有“重置”和“保持”,且带“外部方向”控制信号的双向计数	8	Quad X4 计数器
4	两输入计数(一个加法计数输入信号,一个减法计数输入信号)	9	有“重置”和“保持”控制信号的 Quad X4 计数器

注意：HSC3、HSC4 和 HSC5 只支持 0、2、4、6 和 8 模式。HSC0、HSC1 和 HSC2 支持所有模式。

HSCSTS 数据类型（HSCStsInfo）见表 6-44，它可以显示 HSC 的各种状态，基本上都是只读数据。其中的一些标志可以用于逻辑编程。

表 6-44　HSCSTS 数据类型

参　数	数据类型	描　述
CountEnable	BOOL	使能或停止 HSC 计数
ErrorDetected	BOOL	非零表示检测到错误
CountUpFlag	BOOL	递增计数标志
CountDwnFlag	BOOL	递减计数标志
Mode1Done	BOOL	HSC 是 1(1A)模式或 2(1B)模式,且累加值递增计数至 HP 的值
OVF	BOOL	检测到上溢
UNF	BOOL	检测到下溢
CountDir	BOOL	1:递增计数;0:递减计数
HPReached	BOOL	达到高预设值
LPReached	BOOL	达到低预设值
OFCauseInter	BOOL	上溢导致 HSC 中断
UFCauseInter	BOOL	下溢导致 HSC 中断
HPCauseInter	BOOL	达到高预设值,导致 HSC 中断
LPCauseInter	BOOL	达到低预设值,导致 HSC 中断
PlsPosition	UINT	可编程限位开关(PLS)的位置
ErrorCode	UINT	错误代码,见 HSC 错误代码
Accumulator	DINT	读取累加器实际值
HP	DINT	最新的高预设值设定,可能由 PLS 功能更新
LP	DINT	最新的低预设值设定,可能由 PLS 功能更新
HPOutput	UDINT	最新高预设输出值设定,可能由 PLS 功能更新
LPOutput	UDINT	最新低预设输出值设定,可能由 PLS 功能更新

关于 HSC 状态信息数据结构（HSCSTS）的说明：

在 HSC 执行的周期里，HSC 功能块在“0x01”（HscCmd）命令下 HSC 的状态将会持续更新。

在 HSC 执行的周期里，如果发生错误，错误检测标志将会打开，不同的错误情况对应着如下的错误代码，见表 6-45。

表 6-45　HSC 错误代码

错误代码位	HSC 计数时错误代码	错 误 描 述
15～8(高字节)	0～255	高字节非零表示 HSC 错误由 PLS 数据设置导致； 高字节的数值表示触发错误 PLS 数据中数组编号
7～0(低字节)	0x00	无错误
	0x01	无效 HSC 计数模式
	0x02	无效高预设值
	0x03	无效上溢
	0x04	无效下溢
	0x05	无 PLS 数据

PLS 数据结构（PlsData）

可编程限位开关（PLS）数据是一组数组，每组数组包括高低预设值以及上下溢出值。PLS 功能是 HSC 操作模式的附加设置。当允许该模式操作时（PLSEnable 选通），每次达到一个预设值时，预设和输出数据将通过用户提供的数据更新（即 PLS 数据中下一组数组的设定值）。所以，当需要对同一个 HSC 使用不同的设定值时，可以通过提供一个包含将要使用的数据的 PLS 数据结构达到目的。PLS 数据结构是一个大小可变的数组。注意：一个 PLS 数据体的数组个数不能大于 255。当 PLS 没有使能时，PLS 数据结构可以不用定义。表 6-46 说明每组数组的基本元素作用。

HSC 状态值（Sts 上对应的输出），见表 6-47。

表 6-46　PLS 数据结构元素作用表

命令元素	数据类型	元素描述
字 0～1	DINT	高预设值设置
字 2～3	DINT	低预设值设置
字 4～5	UDINT	高位输出预设值
字 6～7	UDINT	低位输出预设值

表 6-47　HSC 状 态 值

HSC 状态值	状态描述
0x00	无动作(没有使能)
0x01	HSC 功能块执行成功
0x02	HSC 命令无效
0x03	HSC ID 超过有效范围
0x04	HSC 配置错误

另外，在使用 HSC 计数时，注意设置滤波参数，否则 HSC 将无法正常计数。该参数在硬件信息中，如图 6-52 所示，我们使用的是 HSC0，其输入编号是 input0～1。

关于 HSC 中断：

高速计数器一般用于计数达到要求后触发中断，进而处理用户自定义的中断程序。中断的设置在硬件信息中的 Interrupts 中能找到。HSC 中断设置如图 6-53 所示。

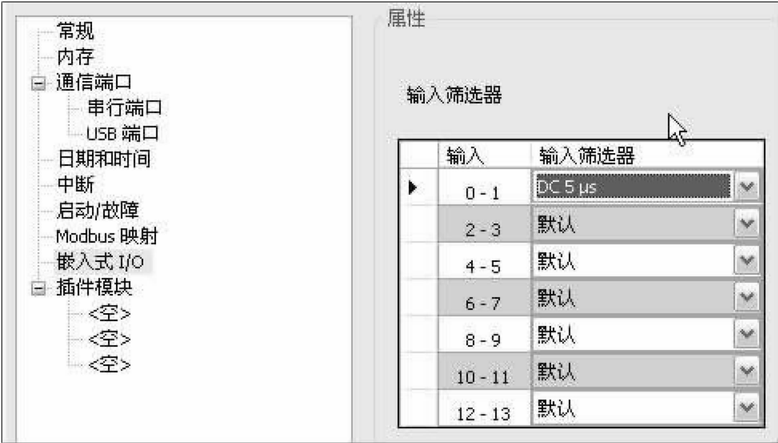


图 6-52　设置滤波参数



图 6-53　HSC 中断设置

我们选择的是 HSC 类型的用户中断，触发该中断的是 HSC0，将要执行的中断程序是 HSCa（用户自定义）。该对话框中我们还看到自动开始参数，当它被置为真时，只要控制器进入任何“运行”或“测试”模式，HSC 类型的用户中断将自动执行。该位的设置将作为程序的一部分被存储起来。“IV 的掩码”表示当该位置假（0）时，程序将不执行检测到的上溢中断命令，该位可以由用户程序设置，且它的值在整个上电周期内将会保持住。类似的“IN 的掩码”、“IH 的掩码”、“IL 的掩码”分别表示屏蔽下溢中断、高设置值中断和低设置值中断。

（2）高速计数器状态设置（HSC _ SET _ STS）

高速计数器状态设置功能块如图 6-54 所示。

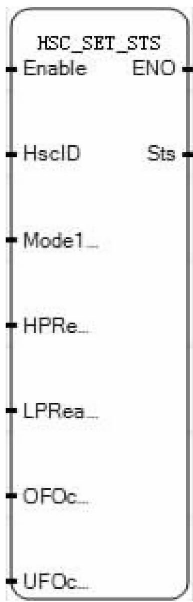


图 6-54　高速计数器状态设置功能块

高速计数器状态设置功能块用于改变 HSC 计数状态。注意：当 HSC 功能块不计数时（停止）才能调用该设置功能块，否则输入参数将会持续更新且任何 HSC _ SET _ STS 功能块做出的设置都会被忽略。

该功能块的参数列表见表 6-48。

表 6-48　高速计数器状态设置功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
HscID	Input	UINT 见 HSC 应用数据结构	欲设置的 HSC 状态
Mode1Done	Input	BOOL	计数模式 1A 或 1B 已完成
HPReached	Input	BOOL	达到高预设值，当 HSC 不计数时，该位可重置为假
LPReached	Input	BOOL	达到低预设值，当 HSC 不计数时，该位可重置为假
OFOccurred	Input	BOOL	发生上溢，当需要时，该位可置为假
UFOccurred	Input	BOOL	发生下溢，当需要时，该位可置为假
Sts	Output	UINT	见 HSC 状态值
ENO	Output	BOOL	使能输出

（3）立即输入（IIM）

立即输入功能块如图 6-55 所示。

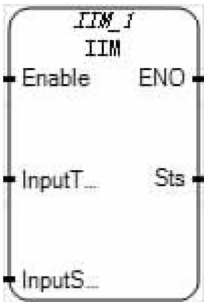


图 6-55　立即输入功能块

该功能块用于不等待自动扫描而立即输入一个数据。注意：对于刚发布的 Connected Components Workbench 版本，IIM 功能块只支持嵌入式的数据输入。

该功能块参数列表见表 6-49。

表 6-49　立即输入功能块参数列表

参　数	参数类型	数据类型	描　述
InputType	Input	USINT	输入数据类型:0——本地数据;1——嵌入式输入;2——扩展式输入
InputSlot	Input	USINT	输入槽号:对于本地输入,总为 0;对于嵌入式输入,输入槽号为 1,2,3,4,5(插口槽号最左边为 1);对于扩展式输入,输入槽号是 1,2,3...(扩展 I/O 模式号,从最左边开始,为 1)
Sts	Output	USINT	立即输入扫描状态,见 IIM/IOM 状态代码

IIM/IOM 状态代码见表 6-50。

4）存储模块信息（MM _ INFO）

表 6-50　IIM/IOM 状态代码

状态代码	描　述
0x00	不使能(不执行动作)
0x01	输入/输出扫描成功
0x02	输入/输出类型无效
0x03	输入/输出槽号无效

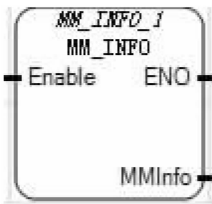


图 6-56　存储模块信息功能块

该功能块用于检查存储模块信息。当没有存储模块时，所有值变为零。其参数列表见表 6-51。

表 6-51　存储模块信息功能块参数列表

参　数	参数类型	数据类型	描　述
MMInfo	Output	MMINFO 见 MMINFO 数据类型	存储模块信息

MMINFO 数据类型见表 6-52。

表 6-52　MMINFO 数据类型

参　　数	数 据 类 型	描　　述
MMCatalog	MMCATNUM	存储模块的目录号,类型编号
Series	UINT	存储模块的序列号,系列
Revision	UINT	存储模块的版本
UPValid	BOOL	用户程序有效(真:有效)
ModeBehavior	BOOL	模式动作(真:上电后,执行运行模式)
LoadAlways	BOOL	上电后,存储模块信息存于控制器
LoadOnError	BOOL	如果上电后有错误,则将存储模块信息存于控制器
FaultOverride	BOOL	上电后出现覆盖错误
MMPresent	BOOL	存储模块信息已存在

（5）嵌入式模块信息（PLUGIN _ INFO）

嵌入式类模块的信息功能块如图 6-57 所示。



图 6-57　嵌入式类模块的信息功能块

嵌入式模块的信息可以通过该功能块读取。该功能块可以读取任意嵌入式模块的信息（除了 2080-MEMBAK-RTC 模块）。当没有嵌入式模块时，所有的参数值归零。其参数列表见表 6-53。

表 6-53　嵌入式模块的信息功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
SlotID	Input	UINT	嵌入槽号:槽号 = 1,2,3,4,5(从最左边开始,第一个插槽号 = 1)
ModID	Output	UINT	嵌入式模块物理 ID
VendorID	Output	UINT	嵌入式模块厂商 ID,对于 Allen Bradley 产品,厂商 ID = 1
ProductType	Output	UINT	嵌入式模块产品类型
ProductCode	Output	UINT	嵌入式模块产品代码
ModRevision	Output	UINT	生产型号版本信息

(6) 嵌入式模块数据读取(PLUGIN _ READ)

嵌入式模块数据读取功能块如图 6-58 所示。

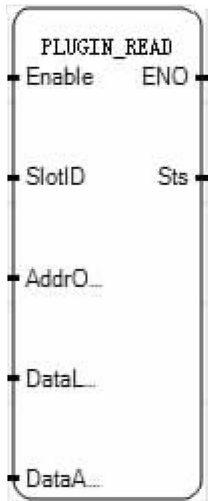


图 6-58 嵌入式模块数据读取功能块

该功能块用于从嵌入式模块硬件读取一组数据。其参数列表见表 6-54。

表 6-54 嵌入式模块数据读取功能块参数列表

参数	参数类型	数据类型	描述
Enable	Input	BOOL	功能块使能。为真时,执行功能块; 为假时,不执行功能块,所有输出数值为 0
SlotID	Input	UINT	嵌入槽号:槽号 =1,2,3,4,5(从最左边开始,槽号 =1)
AddrOffset	Input	UINT	第一个要读的数据的地址偏移量。从嵌入类模块的第一个字节开始计算
DataLength	Input	UINT	需要读的字节数量
DataArray	Input	USINT	任意曾用于存储读取于嵌入类模块 Data 中的数据的数组
Sts	Output	UINT	见嵌入类模块操作状态值
ENO	Output	BOOL	使能输出

嵌入式模块操作状态值见表 6-55。

表 6-55 嵌入式模块操作状态值

状态值	状态描述	状态值	状态描述
0x00	功能块未使能(无操作)	0x03	由于无效嵌入式模块,嵌入操作失败
0x01	嵌入操作成功	0x04	由于数据操作超出范围,嵌入操作失败
0x02	由于无效槽号,嵌入操作失败	0x05	由于数据奇偶校验错误,嵌入操作失败

(7) 嵌入式模块重置(PLUGIN _ RESET)

嵌入式模块重置功能块如图 6-59 所示。

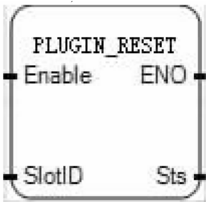


图 6-59　嵌入式模块重置功能块

该功能块用于重置任意嵌入式模块硬件信息（除了 2080-MEMBAK-RTC）。硬件重置后，嵌入式模块可以组态或操作。其参数列表见表 6-56。

表 6-56　嵌入式模块重置功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
SlotID	Input	UINT	嵌入槽号:槽号 =1,2,3,4,5(从最左边开始,槽号 =1)
Sts	Output	UINT	见嵌入式模块操作状态值

（8）读 RTC（RTC _ READ）

读 RTC 功能块如图 6-60 所示。

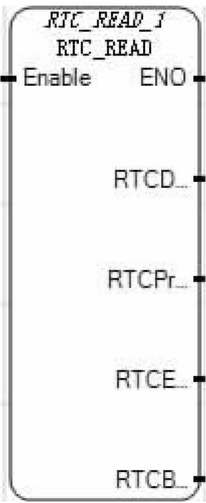


图 6-60　读 RTC 功能块

该功能块用于读取 RTC 预设值和 RTC 信息。

提示：当在带嵌入式的 RTC 的 Micro810 控制器中使用时，RTCBatLow 总是 0。当由于断电导致嵌入式的 RTC 丢失其负载或存储信息时，RTCEnabled 总是为 0。其参数列表见表 6-57。

表 6-57　读 RTC 功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
RTCData	Output	RTC 见 RTC 数据类型	RTC 数据信息:yy/mm/dd,hh/mm/ss,week
RTCPresent	Output	BOOL	真:RTC 硬件嵌入;假:RTC 未嵌入

（续）

参　　数	参数类型	数据类型	描　　述
RTCEnabled	Output	BOOL	真:RTC 硬件使能(计时); 假:RTC 硬件未使能(未计时)
RTCBatLow	Output	BOOL	真:RTC 电量低;假:RTC 电量不低
ENO	Output	BOOL	使能输出

RTC 数据类型见表 6-58。

(9) 写 RTC (RTC _ SET)

写 RTC 功能块如图 6-61 所示。

表 6-58　RTC 数据类型

参　　数	数据类型	描　　述
Year	UINT	对 RTC 设置的年份,16 位,有效范围是 2000 ~ 2098
Month	UINT	对 RTC 设置的月份
Day	UINT	对 RTC 设置的日期
Hour	UINT	对 RTC 设置的小时
Minute	UINT	对 RTC 设置的分钟
Second	UINT	对 RTC 设置的秒
DayOfWeek	UINT	对 RTC 设置的星期

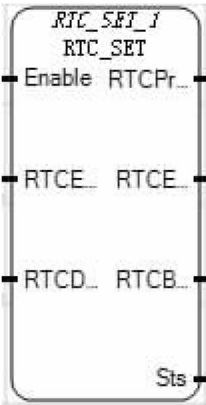


图 6-61　写 RTC 功能块

该功能块用于设置 RTC 状态或是写 RTC 信息。其参数列表见表 6-59。

表 6-59　写 RTC 功能块参数列表

参　　数	参数类型	数据类型	描　　述
RTCEnabled	Input	BOOL	真:使 RTC 能使用 RTC 数据类型;假:停止 RTC 提示:该参数在 Micro810 中忽略
RTCData	Input	RTC 见 RTC 数据类型	RTC 数据信息:yy/mm/dd,hh/mm/ss,week 当 RTCEnabled = 0 时,忽略该数据
RTCPresent	Output	BOOL	真:RTC 硬件嵌入;假:RTC 未嵌入
RTCEnabled	Output	BOOL	真:RTC 硬件使能(定时);假:RTC 硬件未使能(未计时)
RTCBatLow	Output	BOOL	真:RTC 电量低;假:RTC 电量不低
Sts	Output	USINT	读操作状态,见 RTC 设置状态值

RTC 设置状态值见表 6-60。

(10) 系统信息 (SYS _ INFO)

系统信息功能块如图 6-62 所示。

表 6-60　RTC 设置状态值

状态值	状 态 描 述
0x00	功能块未使能(无操作)
0x01	RTC 设置操作成功
0x02	RTC 设置操作失败

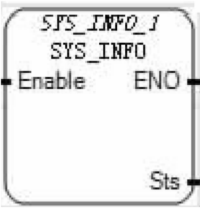


图 6-62　系统信息功能块

该功能块用于读取系统状态数据块。其参数列表见表 6-61。

表 6-61　系统信息功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
Sts	Output	SYSINFO,见 SYSINFO 数据类型	系统状态数据块
ENO	Output	BOOL	使能输出

SYSINFO 数据类型见表 6-62。

表 6-62　SYSINFO 数据类型

参　　数	数 据 类 型	描　　述
BootMajRev	UINT	启动主要版本信息
BootMinRev	UINT	启动副本信息
OSSeries	UINT	操作系统(OS)系列。注:0 代表系列 A 产品
OSMajRev	UINT	操作系统(OS)主要版本
OSMinRev	UINT	操作系统(OS)次要版本
ModeBehaviour	BOOL	动作模式(真:上电后启动 RUN 模式)
FaultOverride	BOOL	默认覆盖(真:上电后覆盖错误)
StrtUpProtect	BOOL	启动保护(真:上电后启动保护程序)。注:对于未来版本
MajErrHalted	BOOL	主要错误停止(真:主要错误已停止)
MajErrCode	UINT	主要错误代码
MajErrUFR	BOOL	用户程序里的主要错误。注:为将来预留
UFRPouNum	UINT	用户错误程序号
MMLoadAlways	BOOL	上电后,存储模块总是重新存储到控制器(真:重新存储)
MMLoadOnError	BOOL	上电后,如果发生错误,则重新存储至控制器(真:重新存储)
MMPwdMismatch	BOOL	存储模块密码不匹配(真:控制器和存储模块的密码不匹配)
FreeRunClock	UINT	从 0 ~ 65535 每 100μs 递增一个数字,然后回到 0 的可运行时钟。如果需要比标准 1ms 更高分辨率的计时器,可以使用该全局范围内可以访问的时钟。注意:仅支持 Micro830 控制器 Micro810 控制器的值保持为 0
ForcesInstall	BOOL	强制安装(真:安装)
EMINFilterMod	BOOL	修改嵌入的过滤器(真:修改)

(11) 微调电位器 (TRIMPOT _ READ)

微调电位器功能块如图 6-63 所示。

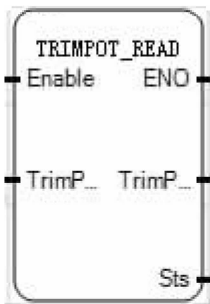


图 6-63　微调电位器功能块

该功能块用于读取微调电位当前值。其参数列表见表 6-63。

表 6-63　微调电位器功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
TrimPotID	Input	UINT	要读取的微调电位的 ID(见 TrimPotID 定义)
TrimPotValue	Output	UINT	当前电位值
Sts	Output	UINT	读取操作的状态(见电位操作状态值)
ENO	Output	BOOL	使能输出

Trimpot ID 定义见表 6-64。

表 6-64　Trimpot ID 定义

输出选择	Bit	描　　述
Trimpot ID 定义	15 ~ 13	电位计模块类型:0x00:本地;0x01:扩展式;0x02:嵌入式
	12 ~ 8	模块的槽号:0x00:本地;0x01 ~ 0x1F:扩展模块的 ID; 0x01 ~ 0x05:嵌入型的 ID
	7 ~ 4	电位类型:0x00:保留;0x01:数字电位类型 1(LCD 模块 1); 0x02:机械式电位计模块 1
	3 ~ 0	模块内部的电位计 ID:0x00 ~ 0x0F:本地;0x00 ~ 0x07:扩展式的电位 ID; 0x00 ~ 0x07:嵌入式的电位 ID。微调电位 ID 从 0 开始

电位操作状态值见表 6-65。

(12) 读校验和 (RPC)

读校验和功能块如图 6-64 所示。

表 6-65　电位操作状态值

状态值	状 态 描 述
0x00	功能块未使能(无读写操作)
0x01	读写操作成功
0x02	由于无效电位 ID 导致读写失败
0x03	由于超出范围导致写操作失败

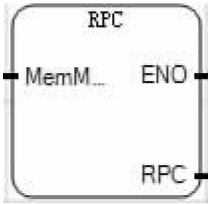


图 6-64　读校验和功能块

用于读取用户程序的校验和，从控制器或者存储模块中。其参数列表见表 6-66。

表 6-66　读校验和功能块参数列表

参　数	参数类型	数据类型	描　述
MemMod	Input	BOOL	为真时,从存储模块中读取; 为假时,从 Micro800 控制器中读取
RPC	Output	UDINT	指定用户程序的校验和
ENO	Output	BOOL	使能输出

8. 中断（Interrupt）

中断功能块指令用途见表 6-67。

表 6-67　中断功能块指令用途

功　能　块	描　述
STIS(定时中断)	用控制程序启动 STI 计时器,而非自动启动
UID(用户中断屏蔽)	屏蔽用户中断
UIE(用户中断使能)	使能用户中断
UIF(用户中断激活)	激活刷新用户中断

（1）定时中断（STIS）

定时中断功能块如图 6-65 所示。

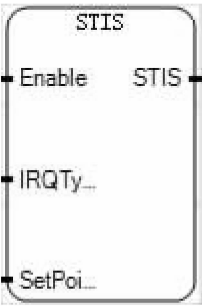


图 6-65　定时中断功能块

用于启动一个可选计时（计时器）用户中断。其参数列表见表 6-68。

表 6-68　定时中断功能块参数列表

参　数	参数类型	数据类型	描　述
IRQType	Input	UDINT	使用 STI 定义字。-IRQ _ STI0;-IRQ _ STI1;-IRQ _ STI2;-IRQ _ STI3
SetPoint	Input	UINT	执行中断前要等待的总时间(单位 ms)。0 表示使 STIS 无效。在 1 ~ 65535 之间的值使能 STIS 功能
STIS	Output	BOOL	梯级状态(与 Enable 一致)

（2）用户中断屏蔽（UID）

用户中断屏蔽功能块如图 6-66 所示。

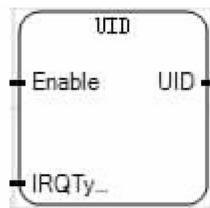


图 6-66　用户中断屏蔽功能块

屏蔽用户中断。其参数列表见表 6-69。

表 6-69　用户中断屏蔽功能块

参　数	参数类型	数据类型	描　述
IRQType	Input	UDINT	使用 IRQ 控制字定义该指令
UID	Output	BOOL	梯级状态(与 Enable 一致)

9. 过程控制（Process Control）

过程控制类功能块指令用途见表 6-70。

表 6-70　过程控制类功能块指令用途

功能块	描　述	功能块	描　述
DERIVATE(微分)	一个实数的微分	IPIDCONTROLLER(PID)	比例,积分,微分
HYSTER(迟滞)	不同实值上的布尔迟滞	SCALER(缩放)	鉴于输出范围缩放输入值
INTEGRAL(积分)	积分	STACKINT(整数堆栈)	整数堆栈

（1）微分（DERIVATE）

微分功能块如图 6-67 所示。

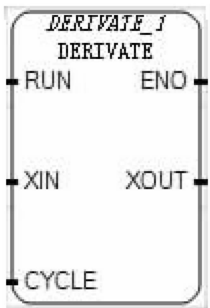


图 6-67　微分功能块

该功能块用于取一个实数的微分。如果 CYCLE 参数设置的时间小于设备的执行循环周期，那么采样周期将强制与该循环周期一致。注意：差分是以毫秒为时间基准计算的（也就是说，求用 1s 时间更改为 2000 的输入 1000 的差分会得到值 1）。要将该指令的输出换算成以秒为单位表示的值，必须将该输出除以 1000。

微分功能块的参数列表见表 6-71。

表 6-71　微分功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
RUN	Input	BOOL	模式:真 = 普通模式;假 = 重置模式
XIN	Input	REAL	输入:任意实数
CYCLE	Input	TIME	采样周期,0 ~ 23h59m59s999ms 之间的任意实数
XOUT	Output	REAL	微分输出
ENO	Output	BOOL	使能输出

（2）迟滞（HYSTER）

迟滞指令功能块如图 6-68 所示。

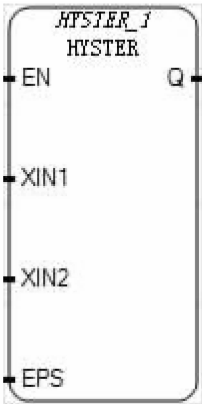


图 6-68　迟滞指令功能块

迟滞指令用于上限实值滞后。其参数列表见表 6-72。

表 6-72　迟滞指令参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
XIN1	Input	REAL	任意实数
XIN2	Input	REAL	测试 XIN1 是否超过 XIN2 + EPS
EPS	Input	REAL	滞后值(须大于零)
ENO	Output	BOOL	使能输出
Q	Output	BOOL	当 XIN1 超过 XIN2 + EPS 且不小于 XIN2-EPS 时为真

该功能块指令的时序图如图 6-69 所示。

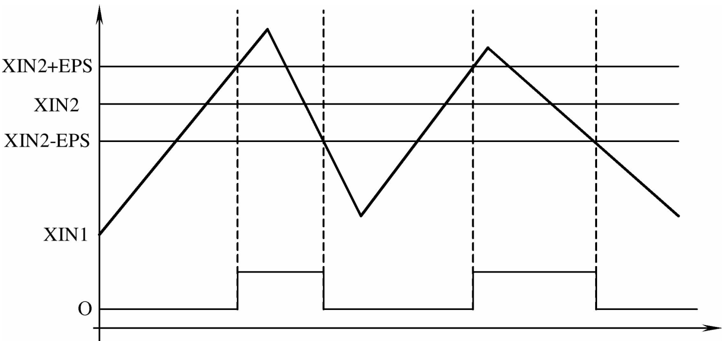


图 6-69　迟滞指令功能块指令的时序图

下面我们来研究一下迟滞指令功能块。从其时序图我们可以看出当功能块输入 XIN1 没有达到功能块的高预置值时（即 XIN2 + EPS），功能块的输出 Q 始终保持 0 状态，当输入超过高预置值时，输出才跳转为 1 状态。输出变为 1 状态后，如果输入值没有小于低预置值（XIN2-EPS），输出将一直保持 1 状态，如此往复。可见迟滞指令功能块是把功能块的输出 1 的条件提高了，然后又把输出 0 的条件降低了。这样的提高启动条件，降低停机条件在实际的应用场合中能起到保护机器的作用。

（3）积分（INTEGRAL）

积分功能块如图 6-70 所示。

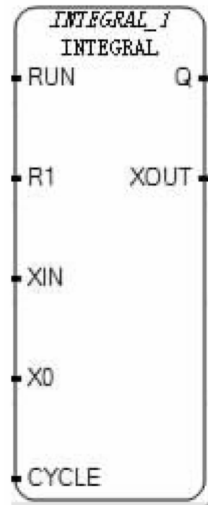


图 6-70　积分功能块

对一个实数进行积分。

提示：如果 CYCLE 参数设置的时间小于设备的执行循环周期，那么采样周期将强制与该循环周期一致。

首次初始化 INTEGRAL 功能块时，不会考虑其初始值。使用 R1 参数来设置要用于计算的初始值。

建议在该功能块不要使用 EN 和 ENO 参数，因为当 EN 为假时循环时间将会中断，导致不正确的积分。如果您选择使用 EN 和 ENO 参数，把 R1 和 EN 置为真来清除现有的结果，以确保积分正确。

为防止丢失积分值，控制器从 PROGRAM 转换为 RUN 或 RUN 参数从“假”转换为“真”时，不会自动清除积分值。首次将控制器从 PROGRAM 转换到 RUN 模式以及启动新的积分时，使用 R1 参数可清除积分值。

该功能块的参数列表见表 6-73。

表 6-73　积分功能块参数列表

参　　数	参数类型	数据类型	描　　述
RUN	Input	BOOL	模式:真 = 积分,假 = 保持
R1	Input	BOOL	重置重写

（续）

参　　数	参数类型	数据类型	描　　述
XIN	Input	REAL	输入:任意实数
X0	Input	REAL	无效值
CYCLE	Input	TIME	采样周期。0 ~ 23h59m59s999ms 间的可能值
Q	Output	BOOL	非 R1
XOUT	Output	REAL	积分输出

（4） PID （IPIDCONTROLLER）

PID 功能块如图 6-71 所示。

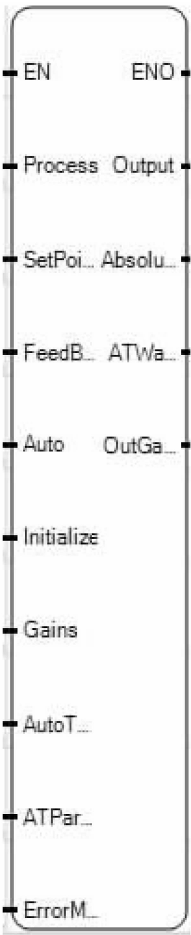


图 6-71　PID 功能块

该功能块为 PID 控制器，它是系统预设的功能块的组合。该功能块的参数列表见表 6-74。

表 6-74　PID 功能块参数列表

参　　数	参数类型	数据类型	描　　述
Process	Input	REAL	过程值,从过程输出中测量
SetPoint	Input	REAL	设定值
FeedBack	Input	REAL	反馈信号,控制变量作用于过程后的值,反馈可以作为 IPIDCONTROLLER 的输出
Auto	Input	BOOL	PID 的操作模式:真:控制器处于正常模式; 假:微分部分将被忽略,且控制器输出将被强制在控制限制的范围内跟踪反馈值,并且允许控制器在没有超过输出限制的条件下跳转到自动调节模式
Initialize	Input	BOOL	初始化。该变量的一个跳变(真变为假或假变为真)将会使控制器在该循环周期中剔除一切比例增益(即初始化)。同时也初始化自动调节(AutoTune)队列
Gains	Input	GAIN _ PID	IPIDController 的 PID 增益,见 GAIN _ PID 数据类型
AutoTune	Input	BOOL	当置为真且 Auto 和 Initialize 为假时,开始执行自动调节阵列
ATParameters	Input	AT _ Param	自动调节的参数配置,见 AT _ Param 数据类型
ErrorMode	Input	DINT	用于处理错误的模式: 0——无错误记录文件(ErrLog file) 1——在错误记录文件中记录级别 1 的错误信息 2——在错误记录文件中记录级别 1 和级别 2 的错误信息
Output	Output	REAL	控制器的输出值
AbsoluteError	Output	REAL	绝对误差(过程值-设定值)
ATWarnings	Output	DINT	自动调节队列的警告: 0——没有自动调节;1——处于自动调节模式;2——已完成自动调节;-1——ERROR 1:控制器的"Auto"信号为真,没有自动调节的可能,请将其置为假;-2——ERROR 2 :自动调节错误,ATDynaSet 到期(即自动调节的等待时间到)
OutGains	Output	GAIN _ PID	自动调节(AutoTune)后计算出的增益

GAIN _ PID 数据类型见表 6-75。

表 6-75　GAIN _ PID 数据类型

参　　数	数据类型	描　　述
DirectActing	BOOL	动作类型: 真:正向操作;假:反向操作
ProportionalGain	REAL	PID 比例增益(≥0. 0001)
TimeIntegral	REAL	PID 积分时间(≥0. 0001)
TimeDerivative	REAL	PID 微分时间(>0. 0)
DerivativeGain	REAL	PID 微分增益(>0. 0)

AT _ Param 数据类型见表 6-76。

表 6-76　AT _ Param 数据类型

参　　数	数据类型	描　　述
Load	REAL	下载自动调节的参数。它是启动 AutoTune 时的输出值
Deviation	REAL	自动调节偏差。这是用于度量自动调节噪声带的标准偏差(噪声带 = 3 * Deviation)
Step	REAL	自动调节的步长,必须大于噪声带的值且小于下载(Load)的 1/2
ATDynamSet	REAL	放弃自动调节前的等待时间(单位为秒)
ATReset	BOOL	决定输出值在自动调节阵列后是否重置为 0 真:自动调节后,重置 IPIDCONTROLLER 输出为 0;假:输出下载值(Load)

如果要运行一个 AutoTune 阵列，ATParameters 输入必须先完成初始设置。Gain 和 DirectActing 参数也必须设定，且 DerivativeGain 也要设置（典型地为 0. 1）。AutoTune 阵列以如下顺序执行：

1）设置初始化输入“Initialize”值为真；

2）设置 Autotune 值为真；

3）改变初始化输入“Initialize”值为假；

4）等待输出 ATWarning 变为 2；

5）从“OutGains”中获取返回值。

为了完成调试，需要一些基于过程和需要的微调。当设置 TimeDerivative 为 0. 0 时，IPIDController 强制 DerivativeGain 为 1. 0，然后变成一个 PI 控制器。

（5）量程转换（SCALER）

量程转换功能块如图 6-72 所示。

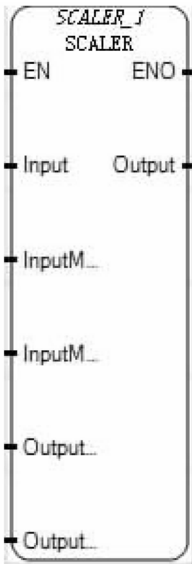


图 6-72　量程转换功能块

基于输出范围量程转换输入值，例如

$$\frac{(Input - InputMin)}{(InputMax - InputMin)} \times (OutputMax - OutputMin) + OutputMin \qquad (6\text{-}1)$$

其参数列表见表 6-77。

表 6-77　量程转换功能块参数列表

参　　数	参数类型	数据类型	描　　述	参　　数	参数类型	数据类型	描　　述
Input	Input	REAL	输入信号	OutputMin	Input	REAL	输出最小值
InputMin	Input	REAL	输入最小值	OutputMax	Input	REAL	输出最大值
InputMax	Input	REAL	输入最大值	Output	Output	REAL	输出值

（6）整数堆栈（STACKINT）

整数堆栈功能块如图 6-73 所示。

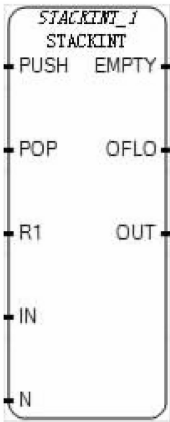


图 6-73　整数堆栈功能块

该功能块用于处理一个整数堆栈。

STACKINT 功能块 PUSH 和 POP 命令的上升沿检测。堆栈的最大值为 128。当重置（R1 至少置为真一次，然后回到假）后 OFLO 值才有效。用于定义堆栈尺寸的 N 不能小于 1 或大于 128。下列情况下，该功能块将处理无效值：

如果 N＜1，STACKINT 功能块尺寸为 1 的数据；

如果 N＞128，STACKINT 功能块尺寸为 128 的数据。

功能块参数列表见表 6-78。

表 6-78　整数堆栈功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
PUSH	Input	BOOL	推命令（仅当上升沿有效），把 IN 的值放入堆栈的顶部
POP	Input	BOOL	拉命令（仅当上升沿有效），把最后推入堆栈顶部的值删除
R1	Input	BOOL	重置堆栈至“空”状态
IN	Input	DINT	推的值

（续）

参　　数	参 数 类 型	数 据 类 型	描　　述
N	Input	DINT	用于定义堆栈尺寸
EMPTY	Output	BOOL	堆栈空时为真
OFLO	Output	BOOL	上溢:堆栈满时为真
OUT	Output	DINT	堆栈顶部的值,当 OFLO 为真时 OUT 值为 0

10. 程序控制（Program Control）

程序控制类功能块指令主要有暂停和限幅以及停止并启动 3 个指令，具体说明如下：

（1）暂停（SUS）

暂停功能块如图 6-74 所示。

该功能块用于暂停执行 Micro800 控制器。其参数列表见表 6-79。

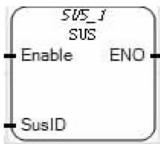


图 6-74　暂停功能块

表 6-79　暂停功能块参数列表

参数	参数类型	数据类型	描　　述
SusID	Input	UINT	暂停控制器的 ID
ENO	Output	BOOL	使能输出

（2）限幅（LIMIT）

限幅功能块如图 6-75 所示。

该功能块用于限制输入的整数值在给定水平。整数值的最大和最小限制是不变的。如果整数值大于最大限值，则用最大限值代替它；小于最小值时，则用最小限值代替它。参数列表见表 6-80。

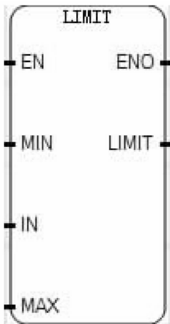


图 6-75　限幅功能块

表 6-80　限幅功能块参数列表

参数	参数类型	数据类型	描　　述
MIN	Input	DINT	支持的最小值
IN	Input	DINT	任意有符号整数值
MAX	Input	DINT	支持的最大值
LIMIT	Output	DINT	把输入值限制在支持的范围内的输出
ENO	Output	BOOL	使能输出

（3）停止并重启（TND）

停止并重启功能块如图 6-76 所示。

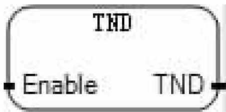


图 6-76　停止并重启功能块

该功能块用于停止当前用户程序扫描，然后在输出扫描、输入扫描和内部处理后，用户程序将从第一个子程序开始重新执行。其参数列表见表 6-81。

表 6-81　停止并重启功能块参数列表

参　　数	参数类型	数据类型	描　　述
TND	Output	BOOL	如果为真,该功能块动作成功。注:当变量监视开启时,监视变量的值将赋给功能块的输出。当变量监视关闭时,输出变量的值赋给功能块输出

6.3.3　函数指令

函数类功能块主要是数学函数，用于快速计算变量之间的数学函数关系。该大类功能块分类及用途见表 6-82。

表 6-82　函数类功能块分类及用途

种　　类	描　　述
算术(Arithmetic)	数学算术运算
二进制操作(Binary operations)	将变量进行二进制运算
布尔运算(Boolean)	布尔运算
字符串操作(String manipulation)	转换提取字符
时间(Time)	确定实时时钟的时间范围,计算时间差

1. 算术（Arithmetic）

算术类功能块指令主要用于实现算术函数关系，如三角函数、指数幂、对数等。该类指令用途见表 6-83。

表 6-83　算术类功能块指令用途

功　能　块	描　　述
ABS(绝对值)	取一个实数的绝对值
ACOS(反余弦)	取一个实数的反余弦
ACOS _ LREAL(长实数反余弦值)	取一个 64 位长实数的反余弦
ASIN(反正弦)	取一个实数的反正弦
ASIN _ LREAL(长实数反正弦值)	取一个 64 位长实数的反正弦
ATAN(反正切)	取一个实数的反正切
ATAN _ LREAL(长实数反正切值)	取一个 64 位长实数的反正切
COS(余弦)	取一个实数的余弦
COS _ LREAL(长实数余弦值)	取一个 64 位长实数的余弦
EXPT(整数指数幂)	取一个实数的整数指数幂
LOG(对数)	取一个实数的对数(以 10 为底)
MOD(除法余数)	取模数
POW(实数指数幂)	取一个实数的实数指数幂
RAND(随机数)	随机值
SIN(正弦)	取一个实数的正弦
SIN _ LREAL(长实数正弦值)	取一个 64 位长实数的正弦
SQRT(平方根)	取一个实数的平方根

（续）

功　能　块	描　　述
TAN（正切）	取一个实数的正切
TAN _ LREAL（长实数正切值）	取一个 64 位长实数的正切
TRUNC（取整）	把一个实数的小数部分截掉（取整）
Multiplication（乘法指令）	两个或两个以上变量相乘
Addition（加法指令）	两个或两个以上变量相加
Subtraction（减法指令）	两个变量相减
Division（除法指令）	两变量相除
1Gain（直接传送）	把一个变量分配到另一个中
Neg（取反）	整数取反

下面举例介绍该类指令的具体应用：

（1） 弧度反余弦值（ACOS）

弧度反余弦值功能块如图 6-77 所示。



图 6-77　弧度反余弦值功能块

该功能块用于产生一个实数的反余弦值。输入和输出都是弧度。其参数列表见表 6-84。

表 6-84　弧度反余弦值功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
IN	Input	REAL	须在 -1.0 ~ 1.0 之间
ACOS	Output	REAL	输入的反余弦值（在 0.0 ~ pi 之间）。无效输入时为 0.0

（2） 除法余数（模）（MOD）

除法余数功能块如图 6-78 所示。

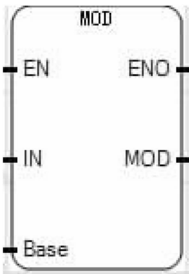


图 6-78　除法余数功能块

用于产生一个整数除法的余数。其参数列表见表 6-85。

表 6-85　除法余数功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
IN	Input	DINT	任意有符号整数
Base	Input	DINT	被除数,须大于零
MOD	Output	DINT	余数计算。如果 Base≤0,则输出 -1

(3) 实数指数幂 (POW)

实数指数幂功能块如图 6-79 所示。

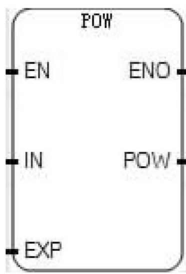


图 6-79　实数指数幂功能块

产生如下形式的实数指数值：基底^{指数} (base^{exponent})。注：exponent 为实数。其参数列表见表 6-86。

表 6-86　实数指数幂功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
IN	Input	REAL	基底,实数
EXP	Input	REAL	指数值,幂
POW	Output	REAL	结果(IN^{EXP}) 输出 1.0,如果 IN 不是 0.0 但 EXP 为 0.0 输出 0.0,如果 IN 是 0.0,EXP 为负 输出 0.0,IN 是 0.0 ,EXP 为 0.0 输出 0.0,如果 IN 为负,EXP 不为整数

(4) 随机数 (RAND)

随机数功能块如图 6-80 所示。

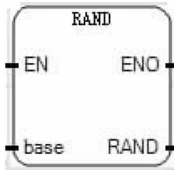


图 6-80　随机数功能块

从一个定义的范围中，产生一组随机整数值。其参数列表见表 6-87。

表 6-87　随机数功能块参数列表

参　数	参数类型	数据类型	描　述
base	Input	DINT	定义支持的数值范围
RAND	Output	DINT	随即整数值，在(0 ~ base - 1)范围内

（5）乘指令（Multiplication）

乘指令功能块如图 6-81 所示。

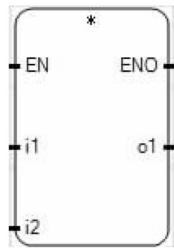


图 6-81　乘指令功能块

两个及多个整数或实数的乘法运算。注意：可以运算额外输入变量。其参数列表见表 6-88。

表 6-88　乘指令功能块参数列表

参　数	参数类型	数据类型	描　述
i1	Input	SINT-USINT-BYTE-INT-UINT-WORD-DINT-UDINT-DWORD-LINT-ULINT-LWORD-REAL-LREAL	可以是整数或实数(所有的输入变量必须是同一格式)
i2	Input		
O1	Output		输入的乘法

（6）直接传送指令（1 gain）

直接传送指令功能块如图 6-82 所示。

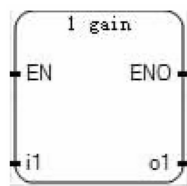


图 6-82　直接传送指令功能块

直接将输入和输出相连接，当与布尔非一起使用时，将一个 i1 复制移动到 o1 中去。其参数列表见表 6-89。

表 6-89　直接传送指令功能块参数列表

参　数	参数类型	数据类型	描　述
i1	Input	BOOL-DINT-REAL-TIME-STRING-SINT-USINT-INT-UINT-UDINT-LINT-ULINT-DATE-LREAL-BYTE-WORD-DWORD-LWORD	输入和输出必须使用相同的格式
o1	Output		输入和输出必须使用相同的格式
ENO	Output	BOOL	使能信号输出

（7）取负指令（Neg）

取负指令功能块如图 6-83 所示。

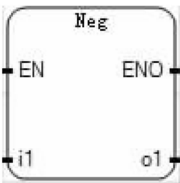


图 6-83　取负指令功能块

将输入变量取反。其参数列表见表 6-90。

表 6-90　取负指令功能块参数列表

参　　数	参数类型	数 据 类 型	描　　述
i1	Input	SINT-INT-DINT-LINT-REAL-LREAL	输入和输出必须有相同的数据类型
o1	Output		

下面通过几个例子讲解一下算术指令的用法。

电动机连续运行时间计时如图 6-84 所示。

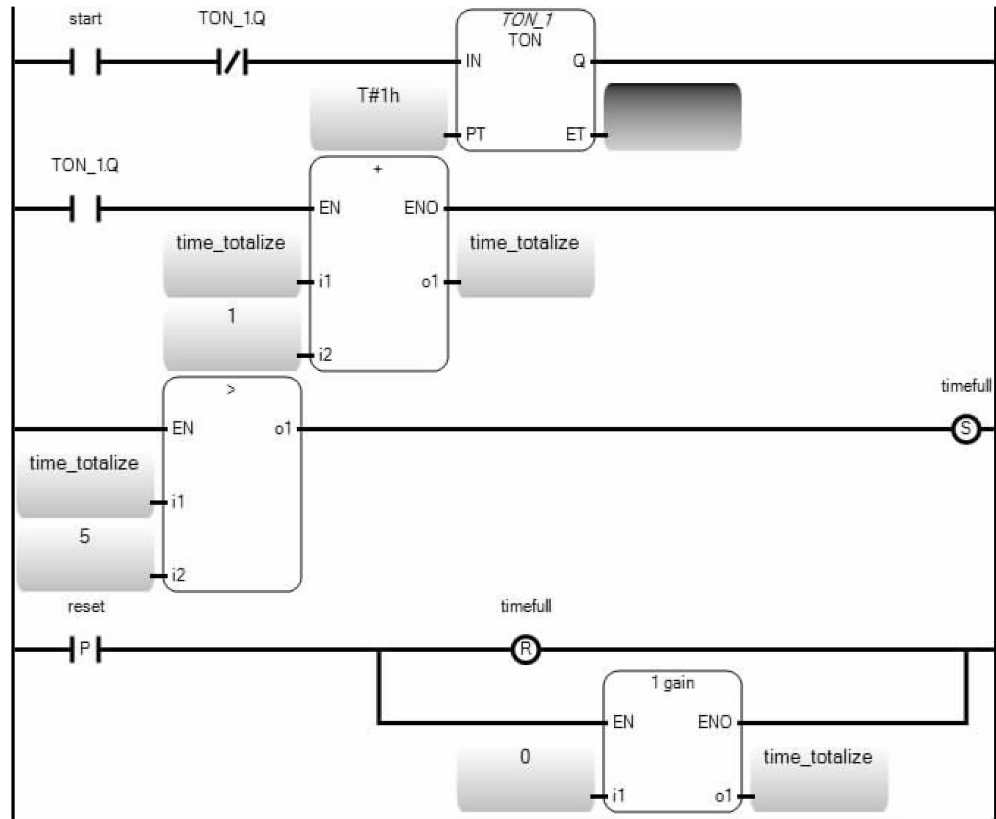


图 6-84　电动机连续运行时间计时

这个程序实现对电动机连续运行时间的计时，用于电动机保养。梯级一是自复位的计时器，循环计时 1h。计时器每计时 1h，通过 TON _ 1. Q 位输出控制 time _ totalize 自加一，当 time _ totalize 大于 5 时，输出 timefull 位。提醒电动机已经连续运行 5h，需要停机。最后一个梯级用于复位 timefull 和 time _ totalize。

定标运算梯级逻辑如图 6-85 所示。



图 6-85　定标运算梯级逻辑

这个程序是定标运算的梯级逻辑。梯级一和梯级二使用 1gain 指令设定未标定范围和标定范围，然后用减法指令计算出未标定范围和标定范围上下限之差，并分别存放到标签 a 和标签 b 中，然后用除法指令计算 b 除以 a，结果存放到标签 k 中。最后输入数据与 k 做乘法，得到定标后的输入值。

2. 二进制操作（Binary Operations）

二进制操作类指令主要用于二进制数之间的与或非运算以及实现屏蔽、位移等功能，该类功能块指令用途见表 6-91。

表 6-91　二进制操作功能块指令用途

功　能　块	描　　述
AND _ MASK（与屏蔽）	整数位到位的与屏蔽
NOT _ MASK（非屏蔽）	整数位到位的取反
OR _ MASK（或屏蔽）	整数位到位的或屏蔽

（续）

功　能　块	描　　述
ROL(左循环)	将一个整数值左循环
ROR(右循环)	将一个整数值右循环
SHL(左移)	将整数值左移
SHR(右移)	将整数值右移
XOR _ MASK(异或屏蔽)	整数位到位的异或屏蔽
AND(逻辑与)	布尔与
NOT(逻辑非)	布尔非
OR(逻辑或)	布尔或
XOR(逻辑异或)	布尔异或

下面将举例介绍该类指令：

（1）取反（NOT _ MASK）

取反功能块如图 6-86 所示。

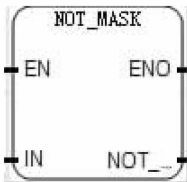


图 6-86　取反功能块

整数值位与位的取反。其参数列表见表 6-92。

表 6-92　取反功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
IN	Input	DINT	须为整数形式
NOT _ MASK	Output	DINT	32 位形式的 IN 的位与位取反
ENO	Output	BOOL	使能输出

例如：16#1234 取 NOT _ MASK 结果为 16#FFFF _ EDCB。

（2）左循环（ROL）

左循环功能块如图 6-87 所示。

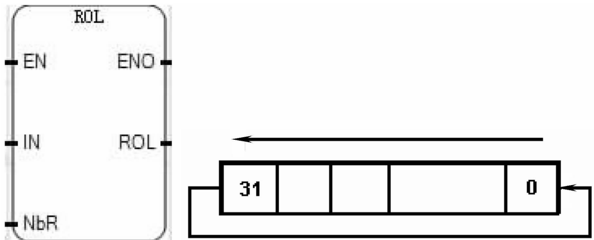


图 6-87　左循环功能块

对于 32 位整数值，把其位向左循环。其参数列表见表 6-93。

表 6-93　左循环功能块参数列表

参　　数	参数类型	数据类型	描　　述
IN	Input	DINT	整数值
NbR	Input	DINT	要循环的位数,须在 1 ~31 范围内
ROL	Output	DINT	左移之后的输出,当 NbR≤0 时,无变化输出
ENO	Output	BOOL	使能输出

(3) 左移(SHL)

左移功能块如图 6-88 所示。

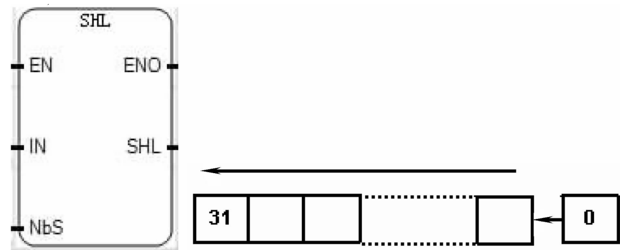


图 6-88　左移功能块

对于 32 位整数值，把其位向左移。最低有效位用 0 替代。其参数列表见表 6-94。

表 6-94　左移功能块参数列表

参　　数	参数类型	数据类型	描　　述
IN	Input	DINT	整数值
NbS	Input	DINT	要移动的位数,须在 1 ~31 范围内
SHL	Output	DINT	左移之后的输出,当 NbR≤0 时,无变化输出

(4) 逻辑与(AND)

逻辑与功能块如图 6-89 所示。

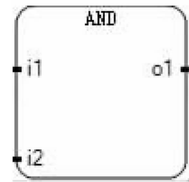


图 6-89　逻辑与功能块

用在两个或更多表达式之间的布尔“与”运算。注意：可以运算额外输入变量。其参数列表见表 6-95。

表 6-95　逻辑与功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
i1	Input	BOOL	
i2	Input	BOOL	
o1	Output	BOOL	输入表达式的布尔与运算

3. 布尔运算（Boolean）

布尔运算功能块指令用途见表 6-96。

表 6-96　布尔运算功能块指令用途

功能块	描　　述
MUX4B	与 MUX4 类似,但是能接受布尔类型的输入且能输出布尔类型的值
MUX8B	与 MUX8 类似,但是能接受布尔类型的输入且能输出布尔类型的值
TTABLE	通过输入组合,输出相应的值

（1）4 选 1（MUX4B）

4 选 1 功能块如图 6-90 所示。

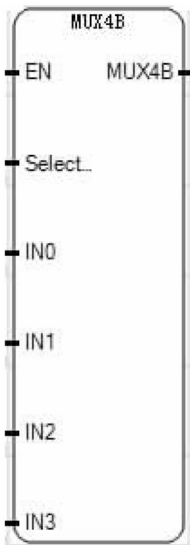


图 6-90　4 选 1 功能块

在 4 个布尔类型的数中选择一个并输出。其参数列表见表 6-97。

表 6-97　4 选 1 功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
Selector	Input	USINT	整数值选择器,须为 0 ~ 3 中的一个值
IN0	Input	BOOL	任意布尔型输入
IN1	Input	BOOL	任意布尔型输入

（续）

参　　数	参数类型	数据类型	描　　述
IN2	Input	BOOL	任意布尔型输入
IN3	Input	BOOL	任意布尔型输入
MUX4B	Output	BOOL	可能为：IN0，如果 Selector = 0；IN1，如果 Selector = 1；IN2，如果 Selector = 2；IN3，如果 Selector = 3 如果 Selector 为其他值时，输出为“假”

（2）组合数（TTABLE）

组合数功能块如图 6-91 所示。

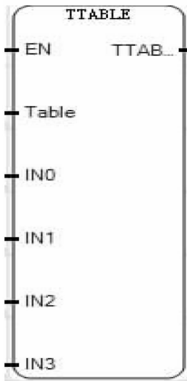


图 6-91　组合数功能块

通过输入的组合，给出输出值。该功能块有 4 个输入，有 16 种组合。可以在真值表中找到这些组合，对于每一种组合，都有相应的输出值匹配。输出数的组合形式取决于输入和该功能块函数的联系。其参数列表见表 6-98。

表 6-98　组合数功能块参数列表

参　　数	参数类型	数据类型	描　　述
Table	Input	UINT	布尔函数的真值表
IN0	Input	BOOL	任意布尔输入值
IN1	Input	BOOL	任意布尔输入值
IN2	Input	BOOL	任意布尔输入值
IN3	Input	BOOL	任意布尔输入值
TTABLE	Output	BOOL	由输入组合而成的输出值

4. 字符串操作（String Manipulation）

字符串操作类功能块指令主要用于字符串的转换和编辑，其用途见表 6-99。

表 6-99　字符串操作功能块指令用途

功　能　块	描　　述
ASCII（ASCII 码转换）	把字符转换成 ASCII 码
CHAR（字符转换）	把 ASCII 码转换成字符
DELETE（删除）	删除子字符串
FIND（搜索）	搜索子字符串
INSERT（嵌入）	嵌入子字符串
LEFT（左提取）	提取一个字符串的左边部分
MID（中间提取）	提取一个字符串的中间部分
MLEN（字符串长度）	获取字符串长度
REPLACE（替代）	替换子字符串
RIGHT（右提取）	提取一个字符串的右边部分

下面将举例介绍该类功能块指令：

（1） ASCII 码转换 （ASCII）

生成 ASCII 功能块如图 6-92 所示。

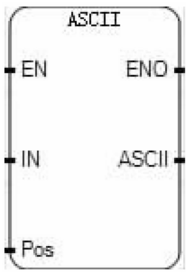


图 6-92　生成 ASCII 功能块

将字符串里的字符变成 ASCII 码。其参数列表见表 6-100。

表 6-100　生成 ASCII 功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
IN	Input	STRING	任意非空字符串
Pos	Input	DINT	设置要选择的字符位置（1 ~ len）（len 是在 IN 中设置的字符串长度）
ASCII	Output	DINT	被选字符的代码（0 ~ 255），若是 0 则 Pos 超出了字符串范围
ENO	Output	BOOL	使能输出

（2） 删除 （DELETE）

删除功能块如图 6-93 所示。

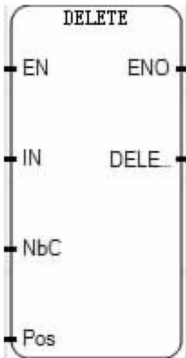


图 6-93　删除功能块

删除字符串中的一部分。其参数列表见表 6-101。

表 6-101　删除功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
IN	Input	STRING	任意非空字符串
NbC	Input	DINT	要删除的字符个数
Pos	Input	DINT	第一个要删除的字符位置(字符串的第一个字符地址是 1)
DELETE	Output	STRING	如下情况之一:1. 已修改的字符串;2. 空字符串(如果 Pos < 1);3. 初始化字符串(如果 Pos > IN 中输入的字符串长度);4. 初始化字符串(如果 NbC≤0)

(3) 搜索 (FIND)

搜索功能块如图 6-94 所示。

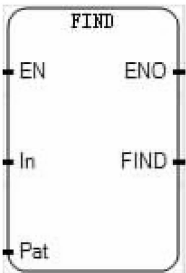


图 6-94　搜索功能块

定位和提供子字符串在字符串中的位置。该功能块的参数列表见表 6-102。

表 6-102　搜索功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
In	Input	STRING	任意非空字符串
Pat	Input	STRING	任意非空字符串(样品 Pattern)

（续）

参　　数	参数类型	数据类型	描　　述
FIND	Output	DINT	可能是如下情况:0:没有发现样品子字符串;子字符串 Pat 第一次出现的第一个字符的位置(第一个位置为 1)
ENO	Output	BOOL	使能输出

（4）左提取（LEFT）

左提取功能块如图 6-95 所示。

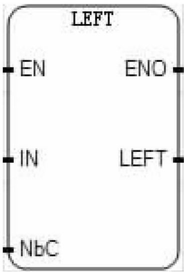


图 6-95　左提取功能块

该功能块用于提取字符串中用户定义的左边的字符个数。其参数列表见表 6-103。

表 6-103　左提取功能块参数列表

参　　数	参数类型	数据类型	描　　述
IN	Input	STRING	任意非空字符串
NbC	Input	DINT	要提取的字符个数,该数不能大于 IN 中输入的字符长度
LEFT	Output	STRING	IN 中输入的字符的左边部分(长度为 NbC 定义的长度),可能为如下情况:空字符串:如果 NbC≤0; 完整的 IN 字符串:如果 NbC≥IN 中字符串的长度
ENO	Output	BOOL	使能输出

5. 时间（Time）

时间类功能块指令主要用于确定实时时钟的年限和星期范围，以及计算时间差。具体用途见表 6-104。

表 6-104　时间类功能块指令用途

功　能　块	描　　述
DOY(年份匹配)	如果实时时钟在年设置范围内,则置输出为真
TDF(时间差)	计算时间差
TOW(星期匹配)	如果实时时钟在星期设置范围内,则置输出为真

下面将举例介绍该类功能块指令的用途：

（1）年份匹配（DOY）

年份匹配功能块如图 6-96 所示。

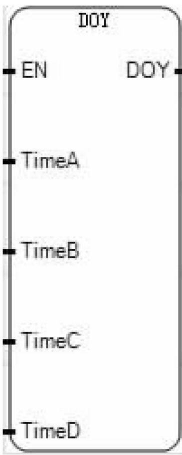


图 6-96　年份匹配功能块

该功能块有 4 个输入通道，当实时时钟（Real-Time Clock，RTC）的值在 4 个通道中任意一个时钟的年份范围内时，功能块输出为真。如果没有 RTC，则输出总为假。其参数列表见表 6-105。

表 6-105　年份匹配功能块参数列表

参　数	参数类型	数据类型	描　述
TimeA	Input	DOYDATA 见 DOYDATA 数据类型	通道 A 的年份设置
TimeB	Input	DOYDATA 见 DOYDATA 数据类型	通道 B 的年份设置
TimeC	Input	DOYDATA 见 DOYDATA 数据类型	通道 C 的年份设置
TimeD	Input	DOYDATA 见 DOYDATA 数据类型	通道 D 的年份设置
DOY	Output	BOOL	真:实时时钟(RTC)的值在 4 个通道中任意一个时钟的年份范围内

DOYDATA 数据类型见表 6-106。

表 6-106　DOYDATA 数据类型

参　数	数据类型	描　述
Enable	BOOL	真:使能;假:无效
YearlyCenturial	BOOL	计时器类型(0:年份计时器;1:世纪计时器)
YearOn	UINT	年的开始值(须在 2000 ~ 2098 之间)
MonthOn	USINT	月的开始值(须在 1 ~ 12 之间)
DayOn	USINT	天的开始值(须在 1 ~ 31 之间,且须与 MonthOn 匹配)
YearOff	UINT	年结束值(须在 2000 ~ 2098 之间)
MonthOff	USINT	月结束值(须在 1 ~ 12 之间)
DayOff	USINT	天结束值(须在 1 ~ 31 之间,且须与 MonthOn 匹配)

6.3.4 运算符

运算符类功能块指令也是 Micro800 控制器的主要指令，该大类指令主要用于转换数据类型以及比较，其中比较指令在编程中占有重要地位，它是一类简单有效的指令。该指令分类见表 6-107。

表 6-107 运算符类功能块指令分类

种　类	描　述
数据转换(Data conversion)	将变量转换为所需数据
比较(Comparators)	变量比较

1. 数据转换（Data Conversion）

数据转换功能块指令主要用于将源数据类型转换为目标数据类型，在整型、时间类型、字符串类型的数据转换时有限制条件，使用时须注意。该类功能块指令用途见表 6-108。

表 6-108 数据转换功能块指令用途

功　能　块	描　述
ANY _ TO _ BOOL(布尔转换)	转换为布尔型变量
ANY _ TO _ BYTE(字节转换)	转换为字节型变量
ANY _ TO _ DATE(日期转换)	转换为日期型变量
ANY _ TO _ DINT(双整型转换)	转换为双整型变量
ANY _ TO _ DWORD(双字转换)	转换为双字型变量
ANY _ TO _ INT(整型转换)	转换为整型变量
ANY _ TO _ LINT(长整型转换)	转换为长整型变量
ANY _ TO _ LREAL(长实型转换)	转换为长实数型变量
ANY _ TO _ LWORD(长字转换)	转换为长字型变量
ANY _ TO _ REAL(实型转换)	转换为实数型变量
ANY _ TO _ SINT(短整型转换)	转换为短整型变量
ANY _ TO _ STRING(字符串转换)	转换为字符串型变量
ANY _ TO _ TIME(时间转换)	转换为时间型变量
ANY _ TO _ UDINT(无符号双整型转换)	转换为无符号双整型变量
ANY _ TO _ UINT(无符号整型转换)	转换为无符号整型变量
ANY _ TO _ ULINT(无符号长整型转换)	转换为无符号长整型变量
ANY _ TO _ USINT(无符号短整型转换)	转换为无符号短整型变量
ANY _ TO _ WORD(字转换)	转换为字变量

下面举例说明该类功能块的应用：

（1）布尔转换（ANY _ TO _ BOOL）

转换成布尔变量功能块如图 6-97 所示。

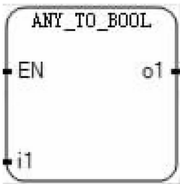


图 6-97 转换成布尔变量功能块

将变量转到布尔变量。其参数列表见表 6-109。

表 6-109　转换成布尔变量功能块参数列表

参　　数	参数类型	数据类型	描　　述
i1	Input	SINT-USINT-BYTE-INT-UINT-WORD-DINT-UDINT-DWORD-LINT-ULINT-LWORD-REAL-LREAL-TIME-DATE-STRING	任何非布尔值
o1	Output	BOOL	可能为:“真”,对于非零数量值而言; “假”,对于零数量值而言; “真”,对于一个“真”字符串而言; “假”,对于一个“假”字符串而言

（2）短整型转换（ANY _ TO _ SINT）

转换成短整型功能块如图 6-98 所示。

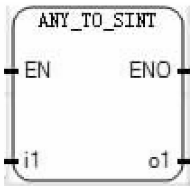


图 6-98　转换成短整型功能块

把输入变量转换为 8 位短整型变量。其参数列表见表 6-110。

表 6-110　转换成短整型功能块参数列表

参　　数	参数类型	数据类型	描　　述
i1	Input	非短整型	任何非短整型值
o1	Output	SINT	计时器的毫秒数,这是一个实数或被字符串代替的小数的整数部分,可能为:“0”,IN 为假 ;“1”,IN 为真
ENO	Output	BOOL	使能信号输出

（3）时间转换（ANY _ TO _ TIME）

转换成时间功能块如图 6-99 所示。

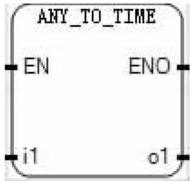


图 6-99　转换成时间功能块

把输入变量（除了时间和日期变量）转换为时间变量。其参数列表见表 6-111。

表 6-111　转换成时间功能块参数列表

参　数	参数类型	数据类型	描　述
i1	Input	见描述	任何非时间和日期变量。IN(当 IN 为实数时,取其整数部分)是以毫秒为单位的数。STRING:毫秒数,例如 300032 代表 5min32ms
o1	Output	TIME	代表 IN 的时间值,1193h2m47s295ms 表示无效输入
ENO	Output	BOOL	使能信号输出

（4）字符串转换（ANY _ TO _ STRING）

转换成字符串功能块如图 6-100 所示。

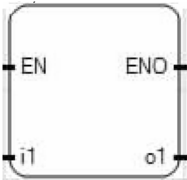


图 6-100　转换成字符串功能块

把输入变量转换为字符串变量。其参数列表见表 6-112。

表 6-112　转换成字符串功能块参数列表

参　数	参数类型	数据类型	描　述
i1	Input	见描述	任何非字符串变量
o1	Output	STRING	如果 IN 为布尔变量,则为“假”或“真” 如果 IN 是整数或实数变量,则为小数 如果 IN 为 TIME 值,可能为: TIME time1;STRING s1;time1:=13ms s1:=ANY _ TO _ STRING(time1);(* s1 = '0s13' *)
ENO	Output	BOOL	使能信号输出

2. 比较（Comparators）

比较功能块指令主要用于数据之间的大小等于比较，是编程时一种简单有效的指令。其用途见表 6-113。

表 6-113　比较功能块指令用途

功　能　块	描　述
Equal(等于)	比较两数是否相等
Greater Than(大于)	比较两数是否其中一个大于另一个
Greater Than or Equal(大于或等于)	比较两数是否其中一个大于或等于另一个
Less Than(小于)	比较两数是否其中一个小于另一个
Less Than or Equal(小于或等于)	比较两数是否其中一个小于或等于另一个

下面举例说明该类功能块的具体应用：

等于（Equal）

等于功能块如图 6-101 所示。

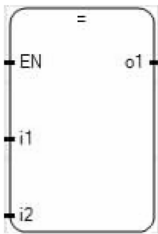


图 6-101　等于功能块

对于整型、实数、时间型、日期型和字符串型输入变量，比较第一个输入和第二个输入，并判断是否相等。其参数列表见表 6-114。

表 6-114　等于功能块参数列表

参　　数	参 数 类 型	数 据 类 型	描　　述
i1	Input	BOOL-SINT-USINT-BYTE-INT-UINT-WORD-DINT-UDINT-DWORD-LINT-ULINT-LWORD-REAL-LREAL-TIME-DATE-STRING	两个输入必须有相同的数据类型。TIME 类型输入只在 ST 和 IL 编程中使用。布尔输入不能在 IL 编程中使用
i2	Input		
o1	Output	BOOL	当 i1 = i2 时为真

提示：由于 TON、TP 和 TOF 功能块作用，不推荐比较 TIME 变量是否相等。

下面通过一个例子讲解比较指令的使用方法。

如图 6-102 所示，这个程序用来控制红灯和蓝灯的亮灭，红灯前 4s 亮，后 4s 灭；蓝灯前 4s 灭，后 4s 亮。梯级一为自复位计时器，用来实现 8s 循环计时。当 TON _ 1. ET 小于等于 4s 时，置位 red，复位 blue。当 TON _ 1. ET 大于 4s 时，置位 blue，复位 red。

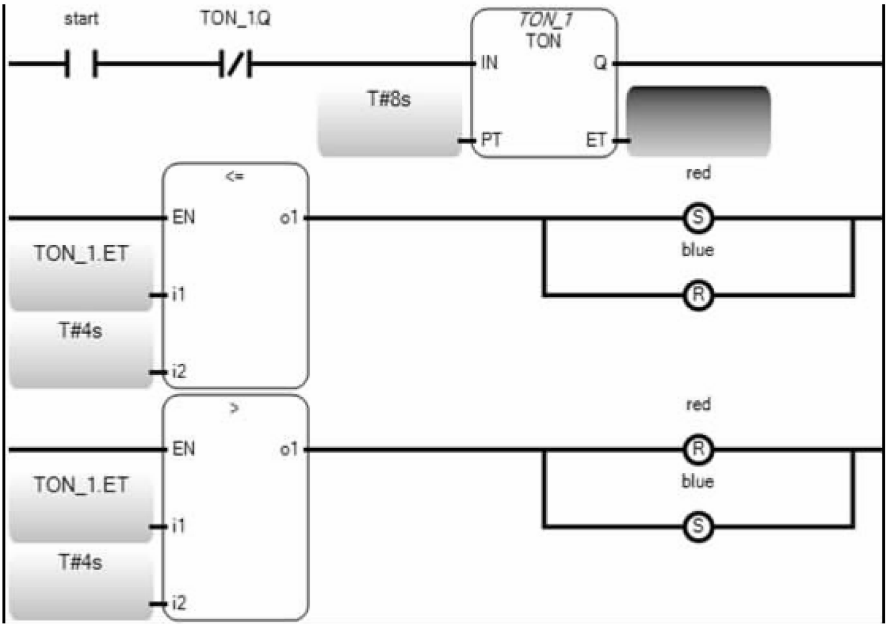


图 6-102　比较指令应用

6.4 小结

PLC 的编程语言种类很多，包括梯形图、结构文本、顺序结构图等。在 Micro800 控制器中最常用的是梯形图语言。Micro800 控制器具有丰富的指令系统，包括：梯形图常用指令、功能块指令、函数指令和操作指令。编程是 PLC 实现工业控制的关键，基本指令的编程是学习编程的基础，当然我们的目的是编出结构更简洁和功能更强大的程序，因此高级指令的学习不可或缺，在后面的章节中就将运用到 MSG 通信指令。这些功能使 Micro800 控制器可以完全满足客户的简洁、经济和实用的要求。

习　题

1. 利用两个定时器和一个输出点设计一个闪烁信号源，使输出的闪烁信号周期为 3s，产生一波形，占空比为 2∶1。

2. 读取逻辑量输入模块的输入状态，记录每个逻辑量输入状态的改变。编写一段中断处理程序来处理引起输入模块输入状态改变的情况（如有硬件出问题了）。

3. 设有一个知识竞赛抢答装置，提出如下控制要求：

主持人用一个开关控制 3 个抢答桌，参赛者若要回答主持人所提问题时需抢先按下桌上的按钮。主持人说出题目后，谁抢先按下桌上的按钮谁的桌上的灯即亮。这时主持人按控制按钮后灯才熄灭，否则一直亮着。3 个抢答桌的按钮作如下安排：一个抢答桌上是儿童组，桌上有两只按钮，并联形式，无论按哪一只，桌上的灯都亮；第二个抢答桌是大学生组，桌子上也有两只按钮，串联形式，只有两只按钮都按下，桌上的灯才亮；第三组是中学生组，桌上只有一只按钮，且只有一个人，一按灯即亮。当主持人将开关置于开状态时，10s 之内若有人按抢答按钮，电铃即发出声响。

4. 控制机械手运动的循环过程为：初始位置、启动、夹紧、正转、松开、反转、回原位、停止。夹紧、松开的时间均为 10s，正转、反转的时间均为 15s，请设计一个逻辑控制程序。

5. 简要描述 MSG 指令的工作机理。

6. 简要描述 HSC 指令的工作机理。

思考题

1. 有 10 个学生，每个学生的数据包括学号、姓名、三门课的成绩，输入 10 个学生的数据，要求计算三门课的平均成绩，分别统计平均成绩高于 80、高于 70 分低于 80 分、高于 60 分而低于 70 分、低于 60 分的学生，并降序排名。

2. 利用顺序进出指令编写交通灯控制程序。

第 7 章

速度控制系统

学习目标

- 了解速度控制系统的结构
- PowerFlex 4M 变频器的 I/O 端子接线
- PowerFlex 4M 变频器的内置键盘操作
- PowerFlex 4M 变频器的 Modbus 网络控制
- 设计速度控制系统

7.1　速度控制系统结构

罗克韦尔自动化控制单元系列中的速度控制解决方案经过优化设计，不但能提供所需要的控制，还能最大程度的降低成本和面板空间。以 Micro830 控制器为核心的速度控制系统的结构如图 7-1 所示。

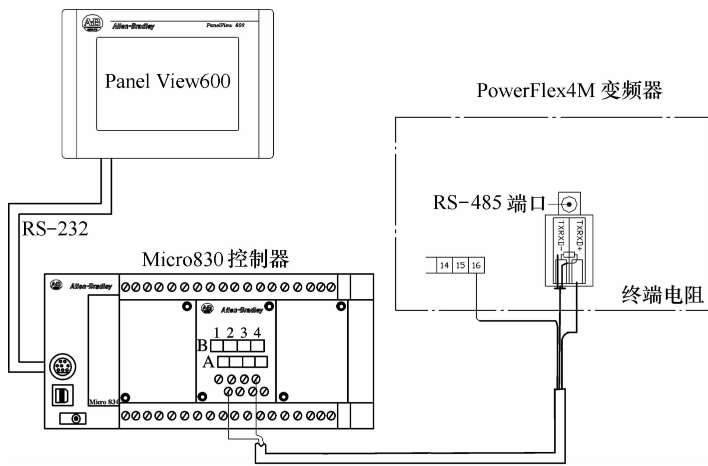


图 7-1　速度控制系统结构

以 Micro830 控制器为核心的速度控制系统采用 PanelView Component 人机界面，可以通过 Modbus 网络进行远程操作。通过预先编辑好的人机界面以及内置状态和报警监控，对控制过程进行设置和调试，并利用低成本的串行网络来控制和监视多台变频器。

速度控制单元的工作原理为：Micro830 控制器经过编程作为 RS-485 的工作主站。控制器通过消息指令（MSG _ MODBUS）子例程与由多台 PowerFlex 4M 变频器组成的网络进行通信，控制每台变频器的启动、停止和方向，监视速度和故障等。PanelView Component 终端通过标准模板为操作员提供控制、状态和诊断屏幕。除此之外，还包括用于电源分配、电路保护、连接性和传感器的组件，以节省成本的方式满足完整控制解决方案的要求。通过 RS-485 和 RS-232 网络，使用智能变频器，有助于在提供应用所需功能的同时，减少控制系统的费用。

Micro830 控制器通过串口与上位机进行通信，将其通道 6 组态为基于 RS-485 的 Modbus 通信协议（参见第 4 章，RS-485 通信），控制变频器的运行，在 Modbus 节点上可以连接多台变频器。选择 PowerFlex 4M 变频器作为驱动设备。速度控制系统单元组件见表 7-1。

表 7-1　速度控制系统单元组件

组件目录号	组件目录描述
2080-LC30-24QWB	Micro830;24V 直流电源,24V 数字输入(14 个),继电器输出(10 个)
1761-CBL-PM02	电缆:将 Micro830 通道 2 的 RS-232 与触摸屏的 RS-232 端口连接

（续）

组件目录号	组件目录描述
22B-D4P0N104	PowerFlex 4M 交流变频器：AC480V，3 相，2.5A，0.75kW，1HP
CM220-FC00118AXMCA	CM220-密封式小型矢量交流电动机：1HP，1800r/min
2711-K6C14	PanelView600 触摸屏终端

速度控制系统的 Micro830 速度控制例程支持与 1～8 个 PowerFlex 4M 变频器进行 Modbus 通信。但是，由于 Modbus 网络通信时一次只能与一台设备进行通信，网络上的设备越多，与全部设备进行通信所需的时间就越长。如果使用默认的通信设置，控制器需要大约 50ms 从每个启用的 PowerFlex 4M 变频器获得状态更新信息，而 PowerFlex 40P 变频器由于需要两个单独的读请求，其更新状态大约需要 100ms。因此，在进行数据通信之前，必须首先确认多台设备的响应时间足够长（即 8 台 PowerFlex 4M 变频器的最大响应时间可达 400ms，8 台 PowerFlex 40P 变频器的最大响应时间可达 800ms）。

8 个变频器通过 RS-485 网络，电缆采用菊花链方式进行连接，并在菊花链的最后一个变频器（仅这个变频器）的连接器上安装终端电阻。但要保证每个变频器都有唯一的节点地址，范围从 1～8。当所有变频器都通电以后，再对它们配置相关参数。

7.2　PowerFlex 4M 交流变频器

PowerFlex 4M 交流变频器是 PowerFlex 变频器系列中最小的一款变频器，它的紧凑、节省空间的设计为用户提供了强大的电动机速度控制功能。它具有使用灵活、馈通式接线和编程简单等特点，是为满足全球 OEM 和终端用户对于灵活性、节省空间和使用方便的要求而设计的理想的元器件级速度控制器。

PowerFlex 4M 交流变频器有三种框架类型（A、B 和 C），可以提供的额定功率为 0.2～11kW，电压等级包括 120V、240V 和 480V。具体特点如下：

（1）安装灵活

可以选择使用 DIN 导轨安装和面板安装；

（2）简易的起动和运行

集成面板可以提供本地电位计和控制键操作直接控制电动机的起动和运行；

数字键盘支持一个 4 位数字显示和 10 个附加的 LED 指示灯，可以直观地显示变频器的状态和信息；

10 个最常用的参数被分为一组以便快速、简便的进行操作；

（3）网络灵活

集成 RS-485 通信，支持多分支网络结构；

使用串行通信转换模块可以连接到任何支持 DF1 协议的控制器端口上；

采用 DriveExplorer 和 DriveExecutive 软件编程、监视和控制变频器；

集成可选用的通信卡，可以提高设备的性能；

（4）优化性能

可拆卸的 MOV 接地，用于不接地供电系统时，可以提供简便操作和无故障运行；

预充电继电器可抑制浪涌电流；

集成的制动晶体管可用于所有额定等级的变频器，使用简单、低成本的制动电阻；

可提供动态制动能力；

可设定 DIP 开关使接线更灵活；

可设置 24V 直流灌入型或拉出型控制，控制接线灵活；

提供强大的过载保护能力：150% 的过载可持续 60s，200% 的过载可以持续 3s；

PWM 频率可调节至 10kHz，保证了静音运行；

V/f 控制性能优良；

变频器可自动进行 IR 补偿和滑差补偿；

整个速度范围内提供卓越的速度控制和高水平的力矩控制，即使在负载增加时也能增强速度调节效果。

7.2.1 PowerFlex 4M 变频器选型

PowerFlex 4M 产品目录号说明如图 7-2 所示。

22F	—	D	018	N	1	0	4	AA
a		*b*	*c*	*d*	*e*	*f*	*g*	*h*

a

变频器	
代码	类型
20F	PowerFlex4M

b

额定电压		
代码	电压	相数
V	交流 120V	1
A	交流 240V	1
B	交流 240V	3
D	交流 480V	3

c1

额定值 100-120V，单相输入		
代码	电流（A）	kW
1P6	1.6	0.2
2P5	2.5	0.4
4P5	4.5	0.75
6P0	6	1.1

c2

额定值 200-240V，单相输入		
代码	电流（A）	kW
1P6	1.6	0.2
2P5	2.5	0.4
4P2	4.2	0.75
8P0	8	1.5
011	11	2.2

c3

额定值 200-240V，三相输入		
代码	电流（A）	kW
1P6	1.6	0.2
2P5	2.5	0.4
4P2	4.2	0.75
8P0	8	1.5
012	12	2.2
017	17.5	3.7
025	25	5.5
033	33	7.5

c4

额定值 380-480V，三相输入		
代码	电流（A）	kW
1P5	1.5	0.4
2P5	2.5	0.75
4P2	4.2	1.5
6P0	6	2.2
8P7	8.7	3.7
013	13	5.5
018	18	7.5
024	24	11

d

机壳	
代码	机壳
N	面板式安装-IP 20 （NEMA 开放式类型）

e

HIM	
代码	HIM 版本
1	固定式嵌入面板

f

辐射等级	
代码	EMC 滤波器
0	无
1	有

g

版本	
代码	描述
3	无制动单元
4	标准内置

h

可选项	
代码	描述
AA 至 ZZ	保留

图 7-2　PowerFlex 4M 产品目录号说明

PowerFlex 4M 产品选型如图 7-3 所示。

120V 交流，单相变频器（50/60Hz）

变频器额定值				IP20，NEMA/UL 开放式
kW	HP	输出电流 A	框架规格	cat. No.
0.2	0.25	1.6	A	22F-V1P6N103
0.4	0.5	2.5	A	22F-V2P5N103
0.75	1	4.5	B	22F-V4P5N103
1.1	1.5	6	B	22F-V6P0N103

240V 交流，单相变频器（50/60Hz）

变频器额定值				IP20，NEMA/UL 开放式	内置 S 型 EMC 滤波器
kW	HP	输出电流 A	框架规格	cat. No.	cat. No.
0.2	0.25	1.6	A	22F-A1P6N103	22F-A1P6N113
0.4	0.5	2.5	A	22F-A2P5N103	22F-A2P5N113
0.75	1	4.2	A	22F-A4P2N103	22F-A4P2N113
1.5	2	8	B	22F-A8P0N103	22F-A8P0N113
2.2	3	11	B	22F-A011N103	22F-A011N113

该滤波器适用于 A 类环境下最长 5 米电缆或 B 类环境下 1 米电缆长度。

240V 交流，三相变频器（50/60Hz）

变频器额定值				IP20，NEMA/UL 开放式
kW	HP	输出电流 A	框架规格	cat. No.
0.2	0.25	1.6	A	22F-B1P6N103
0.4	0.5	2.5	A	22F-B2P5N103
0.75	1	4.2	A	22F-B4P2N103
1.5	2	6	A	22F-B8P0N103
2.2	3	12	B	22F-B012N103
3.7	5	17.5	B	22F-B017N103
内置制动单元				
5.5	7.5	25	C	22F-B025N104
7.5	10	33	C	22F-B033N104

480V 交流，三相变频器（50/60Hz，无滤波）

变频器额定值				IP20. NEMA/UL 开放式	内置 S 型 EMC 滤波器
kW	HP	输出电流 A	框架规格	cat. No.	cat. No.
0.4	0.5	1.5	A	22F-D1P5N103	22F-D1P5N113
0.75	1	2.5	A	22F-D2P5N103	22F-D2P5N113
1.5	2	4.2	B	22F-D4P2N103	22F-D4P2N113
2.2	3	6	B	22F-D6P0N103	22F-D6P0N113
3.7	5	6.7	B	22F-D8P7N103	22F-D8P7N113
内置制动单元					
5.5	7.5	13	C	22F-D013N104	22F-D013N114
7.5	10	18	C	22F-D018N104	22F-D018N114
11	15	24	C	22F-D024N104	22F-D024N114

该滤波器适用于 A 类环境下最长 10 米电缆。

图 7-3　PowerFlex 4M 产品选型

7.2.2 PowerFlex 4M 的 I/O 端子接线

PowerFlex 4M 控制端子接线图如图 7-4 所示。

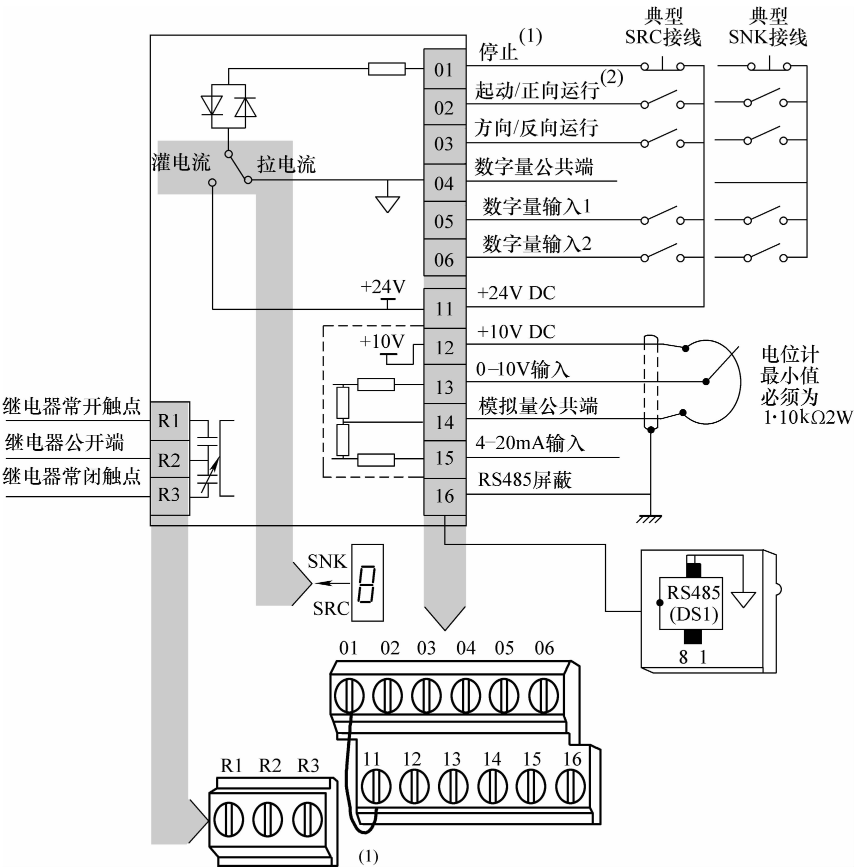


图 7-4 PowerFlex 4M 控制端子接线图

在电动机起动前，用户必须检查控制端子接线：

1）检查并确认所有输入连接是否正确；

2）检查并确认所有的数字量控制电源是 24V；

3）检查并确认灌入（SNK）/拉出（SRC）DIP 开关设置是否正确。

注意：默认状态 DIP 开关为拉出（SRC）状态。I/O 端子 01（停止）和 11（DC +24V）短接以允许从键盘起动。如果控制接线方式改为灌入（SNK），该短接线必须从 I/O 端子 01 和 11 间去掉，并安装到 I/O 端子 01 和 04 之间。各端子说明见表 7-2。

表 7-2 PowerFlex 4M 控制 I/O 端子说明

序 号	信号名称	默认值	说 明	相关参数
R1	常开继电器	故障	输出继电器常开点	A055
R2	继电器公共端	—	输出继电器公共端	
R3	常闭继电器	故障	输出继电器常闭触点	A055

（续）

序　号	信号名称	默认值	说　明	相关参数
灌入/拉出 DIP 开关		拉电流(SRC)	通过 DIP 开关设置,输入端子可接成灌入(SNK)或拉出(SRC)方式	
01	停止	惯性停车	电动机起动前,必须有出厂安装的跳线或常闭输入点	P036
02	起动/正转	未激活	默认状态下,命令来自集成面板控制变频器。如需要禁止反向操作,使用参数 A095【反向禁止】	P036、P037、A095
03	方向/反转	未激活		
04	数字量公共端	—	用于数字量输入	
05	数字量输入 1	预设频率值	通过 A051[数字量输入 1 选择]设定	
06	数字量输入 2	预设频率值	通过 A052[数字量输入 2 选择]设定	
11	DC +24V	—	变频器给数字量输入供电。最大输出电流为 100mA	
12	DC +10V	—	变频器给外部电位计提供 0 ~ 10V。最大输出电流 15mA	
13	0 ~ 10V 输入	未激活	用于外部 0 ~ 10V(输入阻抗 =100kW)或滑动电位计	P038
14	模拟量公共端	—	用于 0 ~ 10V 或 4 ~ 20mA 输入	
15	4 ~ 20mA 输入	未激活	用于外部 4 ~ 20mA 输入供电(输入阻抗 =250kW)	P038
16	RS-485(DSI)屏蔽	—	当使用 RS-485(DSI)通信端口时,需连接到安全地-PE	

7.2.3　PowerFlex 4M 集成式键盘操作

PowerFlex 4M 集成式键盘的外观如图 7-5 所示，菜单说明见表 7-3，各 LED 和按键指示说明见表 7-4、表 7-5。

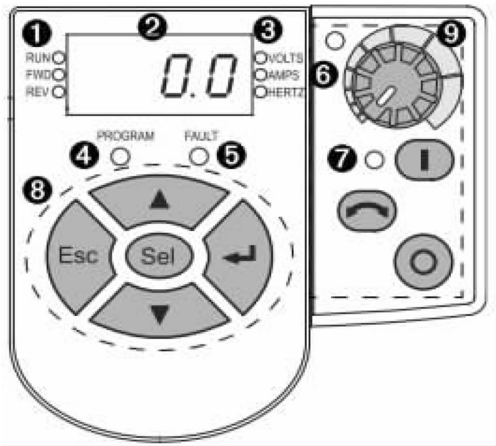


图 7-5　PowerFlex 4M 集成式键盘的外观

表 7-3　菜 单 说 明

菜单	说　　明	菜单	说　　明
d	显示组(只能查看) 包括通常要查看的变频器运行状况	C	通信组 包括通信的可编程功能
P	基本编程组 包括大多数常用的可编程功能	A	高级编程组 包括其余的可编程功能
t	端子组 包括控制端子的可编程功能	F	故障指示 包括特殊故障情况的代码 只有当故障发生时才显示

表 7-4　各指示灯说明

编号	LED	LED 状态	说　　明
1	运行/方向状态	固态红	表示变频器正在运行并且电动机正在按照给定的命令方向运转
		闪烁红	变频器接受命令正在改变方向。当电动机减速到 0 时指示实际电动机方向
2	符号显示	固态红	表示参数号、参数值或故障代码
		闪烁红	单个数字闪烁表示该数字可被编辑,所有数字闪烁表示故障
3	显示单位	固态红	表示当前显示参数的单位
4	编程状态	固态红	表示参数值可以被修改
5	故障状态	闪烁红	表示变频器故障
6	电位计状态	固态绿	表示内置键盘上的电位计处于激活状态
7	起动键状态	固态绿	表示内置键盘上的起动键处于激活状态,且反向禁止【A095】

表 7-5　各按键说明

图　　示	名　　称	说　　明
	电位计	用于控制变频器的转速。默认值为激活。通过参数 P038【速度参考】控制
	起动	用于起动变频器。默认值为激活。通过参数 P038【速度参考】控制
	反转	用于反转变频器方向。默认值为激活。通过参数 P038【速度参考】以及 A095【反向禁止】控制
	停止	用于停止变频器或清除故障。该键一直激活。通过参数 P037【停止模式】控制

熟悉内置键盘各部分的含义后，通过表 7-6 了解如何查看和编辑变频器的参数。

表 7-6　查看和编辑变频器参数

步　　骤	按　　键	显示实例
1. 当变频器上电后，用户上次选择显示组的参数闪烁显示。然后显示该参数当前值		0.0 VOLTS AMPS HERTZ
2. 按“Esc”键，显示上电后的显示组参数。参数号闪烁	Esc	d001 VOLTS AMPS HERTZ
3. 再次按“Esc”键进入参数组菜单	Esc	d001 VOLTS AMPS HERTZ
4. 按向上箭头或向下箭头在主菜单中滚动（d，P 和 A）	▲ or ▼	P101 VOLTS AMPS HERTZ
5. 按“Enter”或选择键进入参数组。该组上次查看的参数最右侧数字将闪烁	↵ or Sel	P101 VOLTS AMPS HERTZ
6. 按向上或向下箭头在组内参数中滚动	▲ or ▼	
7. 按“Enter”或选择键来查看参数值。如果用户不想编辑参数值，按“Esc”键返回	↵ or Sel	230 VOLTS AMPS HERTZ
8. 按“Enter”或选择键进入编程模式来编辑参数值。此时，Program LED 指示灯表示该参数是否可被编辑	↵ or Sel	230 VOLTS AMPS HERTZ PROGRAM FAULT
9. 按向上箭头或向下箭头来修改参数值。达到期望值后，按选择键修改下一位	▲ or ▼ Sel	230 VOLTS AMPS HERTZ PROGRAM FAULT

（续）

步　骤	按　键	显示实例
10. 按“Esc”取消修改，或者按“Enter”键保存修改		
11. 按“Esc”返回到参数列表。连续按“Esc”退出参数菜单		

7.2.4 PowerFlex 4M 的 Modbus 网络通信

PowerFlex 4M 系列变频器集成的 RS-485 通信使其可以在多分支网络结构中使用。简单的 RS-485 通信有以下三种解决方案：

1）PC 通过 DriveExplorer 或 DriveExecutive 软件对变频器进行控制和监视；

2）控制器通过 Modbus 协议与变频器进行通信；

3）与所有可以成为 RTU 主站的设备兼容。

在控制单元速度控制系统中，选择通过 Modbus 协议实现控制器与变频器的通信。

1. Modbus 协议

RS-485/Modbus 是现在流行的一种布网方式，其特点是操作简单、方便，具有 Modbus 接口的设备可以很方便地进行组态。从其功能上看，它可认为是一种现场总线，通过 24 种总线命令实现控制器与外界的信息交换。

Modbus 有两种传送模式，RTU（Remote Terminal Unit）和 ASCII 码。它把通信参与者规定为主站和从站。主站可向多个从站发送通信请求，最多可达 247 个从站，每个从站都有自己的地址编号。对于每一种传输方式，Modbus 协议都定义了控制器可识别和使用的信息类型，而无须考虑通信网络的拓扑结构。即通过定义传输的数据信息帧格式，描述控制器如何访问其他设备和其他设备怎样做出响应，并检查和报告错误的过程。

在该速度控制系统中，采用 Modbus 的 RTU 模式，控制器与设备之间的通信主要包括主站对从站的读取和写入。主站可单独与从站进行通信，也可以广播方式与所有从站进行通信。Modbus 规定，只有主站具有主动权，从站只能被动地响应，包括回答出错信息。主站写入信息的帧格式见表 7-7，从站读取信息的帧格式见表 7-8。

表 7-7　主站写入信息帧格式

从站地址	功能代码	数据量	命令数据	CRC 校验码
1 个字节	1 个字节	1 个字节	N 个字节	2 个字节

表 7-8　从站读取信息帧格式

从站地址	功能代码	数据量	响应数据	CRC 校验码
1 个字节	1 个字节	1 个字节	N 个字节	2 个字节

Modbus 的 RTU 模式规定，采用 CRC（循环冗余校验）方法对整个信息帧的内容进行错误检测，传输信息的最后两个字节用于传递该循环冗余校验数据。其校验方式是将整个传输数据（不包括最后两个字节）的所有字节按规定的方式进行位移并进行 XOR（异或）计算，其结果作为检验码。接收者在收到数据时按同样的方式进行计算，并将结果与收到的 CRC 校验码进行比较，如果一致则认为通信正确，如果不一致，则认为通信有误，从站将发送 CRC 错误响应。

Modbus 通信包括 24 种功能命令，每一种功能命令都有相应的功能代码。最基本的功能包括模拟量输入/输出（AI/AO）和数字量输入/输出（DI/DO），控制器如果支持 Modbus 都应该包含这些基本命令，并将模拟量信息存放在寄存器（Holding Register）中，数字量信息存放在线圈（Holding Coils）中。

2. PowerFlex 4M 变频器的 Modbus 功能代码

PowerFlex 4M 变频器的外设接口（DSI）支持部分 Modbus 功能代码，Modbus 功能代码和命令见表 7-9。

表 7-9　Modbus 功能代码和命令

Modbus 功能代码(十进制)	命　令
03	读寄存器
06	写单个寄存器
16	写多个寄存器

通过 Modbus 协议，控制器向 PowerFlex 4M 变频器的寄存器中写入逻辑命令及速度给定信息，其相应的寄存器地址参见第 4 章 RS-485 通信；控制器从 PowerFlex 4M 变频器寄存器中读取逻辑状态及速度反馈信息，其相应的寄存器地址也参见第 4 章 RS-485 通信一节。下面强调几点要注意的地方：

● 寄存器地址偏移量为 1，例如：逻辑命令的寄存器地址是 8192，而实际操作中就要设置为 8193。

● 通过 Modbus 网络控制变频器，因此 PowerFlex 4M 的参数 P036［Start Source］（起动源）和 P038［Speed Reference］（速度参考）都设为 5——“通信端口”。

● 控制器可通过发送功能代码 06 将控制信息写入地址为 8193（逻辑命令字）和 8194（速度给定值）的寄存器中，以控制变频器的运行。也可通过发送功能代码 03，读取地址为 8449（逻辑状态字）和 8452（速度反馈值）的寄存器中的信息。

读写变频器其他参数时，寄存器地址就是相应的参数号码，但是要注意偏移 1 位。

3. Modbus 网络的硬件连接

本系统使用 Micro830 控制器，在 RS-485 网络上通过 Modbus RTU 模式监视并控制 PowerFlex 4M 系列变频器。硬件接线见本章第一节。

4. PowerFlex 4M 变频器参数设置

通过变频器控制面板设置参数，控制面板操作参见表 7-6，需要设置的参数如下：

1） P036［Start Source］设为 5——“Comm Port”（通信端口）；

2） P038［Speed Reference］设为 5——“Comm Port”（通信端口）；

3） A051［数字量输入 1 选项］设为 6——“通信端口”；

4）A052［数字量输入 2 选项］设为 0——“不使用”；

5）A103［通信数据传输率］设为 4——“19.2K”；

6）A104［通信节点地址］设为 100；

7）A107［通信格式］设置为 0——“RTU 8-N-1”。

5. Micro830 控制器组态

与第 4 章 RS-485 通信一节中介绍的一样，设置好变频器的参数后，即可对控制器进行编程，且在程序中不需要组态变频器，直接通过 MSG _ MODBUS 指令即可控制变频器。

1）将嵌入式串口通信模块插到控制器的 2 槽上，把该模块组态成 Modbus 网络协议（则该模块的通道号为 6 号，详见第 6 章 Modbus 指令），通信驱动类型（Driver）设置为 Modbus RTU Master。打开 Micro830 控制器的组态界面，选择对 2 槽上的串口通信模块进行组态，设置以下参数：

- Driver（通信驱动类型）：Modbus RTU；
- Baud Rate（通信波特率）：19200；
- Party（奇偶校验位）：NONE；
- Unit Address（控制器节点地址）：1；
- Modbus Role（在 Modbus 中的角色）：Modbus RTU Master。

2）单击 Advanced Settings 按钮，展开通信模块的高级设置，设置如下参数：Media（协议类型）：RS-485；Stop Bites（数据格式停止位）：1。

6. Micro830 控制器与 PowerFlex 4M 变频器的通信程序

1）创建 MSG _ MODBUS 功能块，并分别创建功能块上所需要的变量，如图 7-6 所示。

2）编写读取变频器逻辑状态字的程序如图 7-7 所示。

名称	数据类型
MSG_MODBUS_1	MSG_MODBUS
cancel	BOOL
D1_lcfg	MODBUSLOCPARA
D1_tcfg	MODBUSTARPARA
D1_laddr	MODBUSLOCADDR
errorID	UINT
error	BOOL

图 7-6　建立 MSG 功能块和它所对应的变量

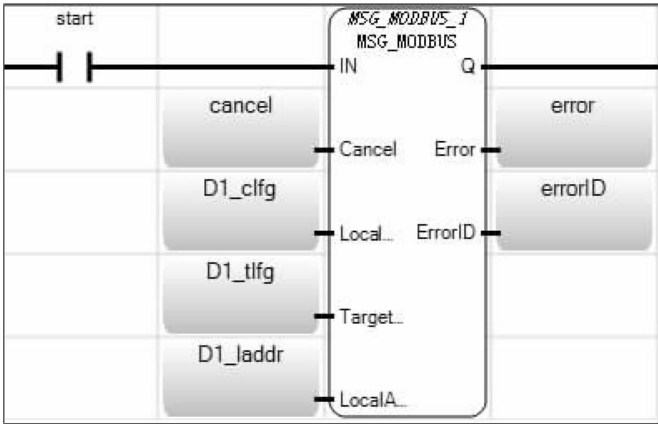


图 7-7　读取变频器逻辑状态字的程序

其中，梯级中的 MSG _ MODBUS _ 1 指令用于读取变频器的逻辑状态字，start 指令用于启动指令，当 start 指令由假变真一次，控制器就会读一次变频器的逻辑状态字。

读取变频器逻辑状态的 MSG 指令参数设置如图 7-8 所示，参数设置见表 7-10。

名称	初始值
MSG_MODBUS_1	...
cancel	
D1_lcfg	...
D1_lcfg.Channel	6
D1_lcfg.TriggerType	0
D1_lcfg.Cmd	3
D1_lcfg.ElementCnt	4
D1_tcfg	...
D1_tcfg.Addr	8449
D1_tcfg.Node	100
D1_laddr	...
D1_laddr[1]	
D1_laddr[2]	
D1_laddr[3]	
D1_laddr[4]	

图 7-8　读取变频器逻辑状态的 MSG 指令参数设置

表 7-10　读取变频器逻辑状态 MSG 指令参数设置

名　　称	作　　用	设　定　值
D1 _ lcfg. Channel(通道)	选择通信端口	6
D1 _ lcfg. Trigger Type(触发类型)	选择触发类型	0(上升沿触发)
D1 _ lcfg. cmd(Modbus 命令)	选择信息功能	3(读寄存器)
D1 _ lcfg. ElementCnt(长度)	选择读取的数据个数	4
D1 _ tcfg. Addrs(Modbus 数据地址)(1-65535)	选择变频器的数据寄存器地址	8449(变频器内部定义)
D1 _ tcfg. Node(从节点地址)	选择变频器的节点地址	100(在 A[104]通信节点地址中设定)
D1 _ laddr[1] ~ D1 _ laddr[4](存放数据的地址)	分别存放从 Modbus 地址 8449 ~8452 中读取的数据	

从 Modbus 地址 8449 ~8452 中读取的数据分别放到 D1 _ laddr[1] ~ D1 _ laddr[4]中，其中 8449 中是变频器逻辑状态字，8450 中是变频器错误代码，8451 中是变频器速度参考值，8452 中是变频器速度反馈值。但是值得注意的是，这里的 Modbus 地址都是经过偏移一位以后的地址。

3）编写控制变频器逻辑命令字的程序与读取逻辑状态字类似，只是 MSG 文件不同，且 MSG _ MODBUS 指令的相关参数设置也有所不同，Modbus 命令选择为“6”，存放数据的地址为“D2 _ laddr[1]”，将该地址文件中的数据写入到变频器寄存器中，而 Modbus 数据地

址（变频器数据寄存器地址）为“8193”，D2 _ laddr 设为 18，命令电动机起动并正转，如图 7-9 所示。

程序编写完成后，将变频器运行位设为 1 时变频器起动，且读取的状态反馈字中的运行位为 1 表示变频器为运行状态。

4）编写设定速度给定值的程序与编写逻辑命令字类似，只是 MSG 文件不同，且 MSG _ MODBUS 指令的相关参数设置也有所不同，Modbus 命令选择为“6”，存放数据的地址为“D3 _ laddr[1]”，将该地址文件中的数据写入到变频器寄存器中，而 Modbus 数据地址（变频器数据寄存器地址）为“8194”，如图 7-10 所示。

名称	初始值
MSG_MODBUS_2	...
cancel2	
D2_lcfg	...
D2_lcfg.Channel	6
D2_lcfg.TriggerType	0
D2_lcfg.Cmd	6
D2_lcfg.ElementCnt	1
D2_tcfg	...
D2_tcfg.Addr	8193
D2_tcfg.Node	100
D2_laddr	...
D2_laddr[1]	18
D2_laddr[2]	

图 7-9 控制变频器逻辑命令字的 MSG _ MODBUS 指令参数设置

名称	初始值
MSG_MODBUS_3	...
cancel3	
D3_lctg	...
D3_lctg.Channel	6
D3_lctg.TriggerType	0
D3_lctg.Cmd	6
D3_lctg.ElementCnt	1
D3_tcfg	...
D3_tcfg.Addr	8194
D3_tcfg.Node	100
D3_laddr	...
D3_laddr[1]	500

图 7-10 设定速度给定值的 MSG _ MODBUS 指令参数设置

5）变频器其他参数的修改与读取都可以用 MSG _ MODBUS 指令来实现，此时寄存器的地址就是相应的参数号码（注意偏移 1 位）。例如，要修改变频器参数 P039 ［Accel time1］，则将寄存器地址设置为 040。

7.3 速度控制系统设计

本节主要讲述速度控制系统的程序设计。整个速度控制系统的控制器程序设计主要包括以下两个部分：

1）变频器控制（DRIVE CTRL），它是整个程序设计最核心的部分。

2）PanelView Component 控制（PVC CTRL）。

在编写程序之前首先要完成控制器、变频器和触摸屏之间在通信时所需要的设置，这一内容在本书的第 4 章以及本章 7.2.4 节中有详细的介绍，这里就不再赘述。下面首先介绍本程序的功能：通过触摸屏来控制 Micro830 控制器向变频器发号施令，从而控制电动机的运行方式和运行速度。下面介绍程序的编写。

1. 变频器控制

首先建立编写程序所需要的变量，速度控制系统程序变量列表见表7-11。变量建完以后，按照前面章节介绍的方法创建程序，并按照表7-11建立MSG _ MODBUS指令梯级，由于这些内容在前面的控制器通信中已经介绍过了，这里不再赘述。

表7-11　速度控制系统程序变量列表

变　　量	作　　用	设　定　值
cancle1	MSG _ MODBUS _ 1 指令的各项参数，用来完成读变频器状态字、错误代码和速度反馈的值	设置为控制器通信端口6工作，触发方式为0，Modbus指令为3，读取数据的长度为4，读取的Modbus首地址为8449，变频器节点地址为100
error1		
errorID1		
D1 _ lcfg		
D1 _ tcfg		
D1 _ laddr		
cancle2	MSG _ MODBUS _ 2 指令的各项参数，用来给变频器写速度30.0Hz	设置为控制器通信端口6工作，触发方式为0，Modbus指令为6，写的数据的长度为1，写的Modbus地址为8194，变频器节点地址为100，速度给定值为300，即30.0Hz
error2		
errorID2		
D2 _ lcfg		
D2 _ tcfg		
D2 _ laddr		
cancle3	MSG _ MODBUS _ 3 指令的各项参数，用来启动变频器	设置为控制器通信端口6工作，触发方式为0，Modbus指令为6，写的数据的长度为1，写的Modbus地址为8193，变频器节点地址为100，变频器控制字设为18
error3		
errorID3		
D3 _ lcfg		
D3 _ tcfg		
D3 _ laddr		
cancle4	MSG _ MODBUS _ 4 指令的各项参数，用来停止变频器	设置为控制器通信端口6工作，触发方式为0，Modbus指令为6，写的数据的长度为1，写的Modbus地址为8193，变频器节点地址为100，变频器控制字设为1
error4		
errorID4		
D4 _ lcfg		
D4 _ tcfg		
D4 _ laddr		
start	送启动变频器控制字	初始值为0
stop	送停止变频器控制字	初始值为0
read	读取变频器各个字	初始值为0
write	给变频器速度	初始值为0
TON _ 1	构成连续读取变频器各个字的梯级	定时3s
TON _ 2		定时3s
light		初始值为0
Logic _ status _ word	变频器状态字	初始值为0

（续）

变　　量	作　　用	设　定　值
error _ code	变频器错误代码	初始值为 0
Speed _ Feedback	变频器速度反馈	初始值为 0
Logic _ Command _ word	变频器命令字	初始值为 0
Speed _ reference _ word	变频器速度给定值	初始值为 0

唯一需要介绍的梯级是持续读取更新变频器各个字的梯级。如图 7-11 所示，计时器 1 计时 3s 后，read 变为真，读取变频器的各个字，3s 后计时器 2 开始计时，计时 3s，light 变为 1，断开梯级，read 和 light 变为假，同时计时器 1 开始计时，3s 后 read 再次变为真，这样就实现了持续的读取更新变频器各个字。

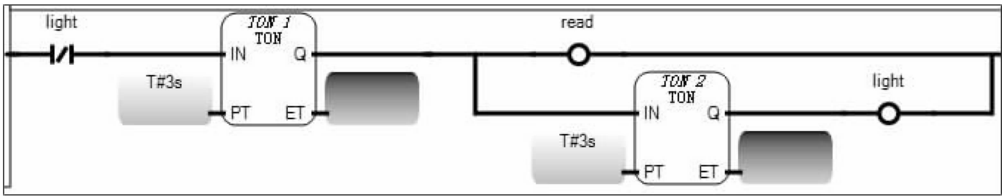


图 7-11　持续读取更新变频器各个字的梯级

2. 触摸屏控制

控制器要和触摸屏通信，就要用到 MODBUS 地址，所以在和触摸屏通信之前，首先要做好 MODBUS 标签地址的映射。由于 MSG _ MODBUS 指令中用的变量不能直接映射 MODBUS 地址，所以首先要用 1gain 指令把 MSG _ MODBUS 指令中的变量传送到可以映射 MODBUS 地址的变量中。例如：把 D1 _ laddr［1］中的变频器状态字传送到变量 Logic _ status _ word 中，如图 7-12 所示。用同样的方法，分别把变频器错误代码、速度反馈、控制字和速度给定值也传送到变量 error _ code、Speed _ Feedback、Logic _ Command _ word 和 Speed _ reference _ word 中。

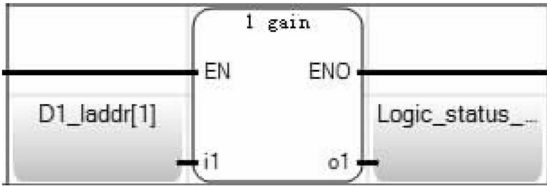


图 7-12　1gain 指令的使用

完成程序中的变量传送以后，就要开始做 MODBUS 地址映射了。在控制器组态界面中，选择 Modbus 映射，如图 7-13 所示。在右边区域中做如图 7-14 所示的 MODBUS 地址映射。这里变量的名字是由“原变量的名字@变量所在程序名”组成的。Modbus 地址映射的规则在第 4 章第 4 节中已经介绍过了，这里不再赘述。需要注意的是，由于在触摸屏中希望看到变频器的状态字和控制字的每个位，所以这里把变量类型为 WORD 的 Logic _ Command _ word 和 Logic _ status _ word 变量映射到布尔量的 MODBUS 地址中，其中地址栏中显示的是映射的首地址，使用地址栏中显示的是整个字所对应的地址范围。如果想要在程序中使用这两个字的某一位，例如：使用

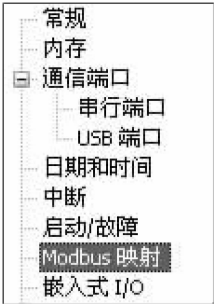


图 7-13　选择 Modbus 映射

Logic _ Command _ word 字中的第 0 位，则写成 Logic _ Command _ word. 0，使用第 1 位则写成 Logic _ Command _ word. 1。

变量名	数据类型	地址	已用地址
start@speed_control_last	Bool	000001	000001
stop@speed_control_last	Bool	000002	000002
write1@speed_control_last	Bool	000003	000003
write2@speed_control_last	Bool	000004	000004
Logic_Command_word@speed_control_last	Word	000005	000005 - 000020
Logic_status_word@speed_control_last	Word	100001	100001 - 100016
Speed_Feedback@speed_control_last	Word	400007	400007
Speed_reference_word@speed_control_last	Word	400008	400008
error_code@speed_control_last	Word	400006	400006

图 7-14　Modbus 地址映射

以上就完成了整个速度控制系统的控制器程序设计。下面介绍在触摸屏编程软件 Panel-Builder32 中的编程。编程之前的通信设置在第五章中有介绍，这里直接介绍触摸屏程序的编写。首先建立标签，打开标签编辑器，如图 7-15 所示。建立需要的变量，如图 7-16 所示。建立变量，首先要选择变量类型，这里用到的变量有三种类型：Ouput Coil（对应 Modbus 地址为 0xxxxx）、Input Status（对应 Modbus 地址为 1xxxxx）和 Holding Registe（对应 Modbus 地址为 4xxxxx）。选择好对应的变量类型以后，要给变量选择相应的地址，例如：启动变频器变量的 Modbus 地址为 000001，这里选择变量类型为 Output Coil，变量地址为 1；状态字的第 0 位对应的 Modbus 地址为 100001，则选择变量类型为 Input Status，变量地址为 1；速度反馈变量的 Modbus 地址为 400007，则选择变量类型为 Hoding Registe，变量地址为 7。按此方法建立其他所需要的变量即可。



图 7-15　标签编辑器

Tag Name	Type	Address	Data Type	Node Name	Initial Value
start	Ouput Coil	1	Bit	speed_control	0
status_0	Input Status	1	Bit	speed_control	0
Speed_Feedback	Holding Registe	7	Unsigned Integer	speed_control	0

图 7-16　建立编程所需变量

建立完所需要的标签，就可以开始编写触摸屏画面了。项目创建时，默认有一个画面，把此画面命名为 speed control。打开画面属性对话框，如图 7-17 和图 7-18 所示，设置画面的各项属性，包括名字、背景颜色、画面描述和网格大小等。

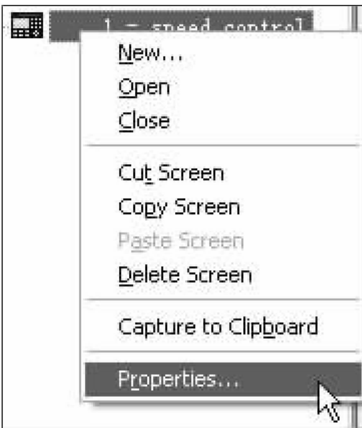


图 7-17　选择打开画面属性对话框

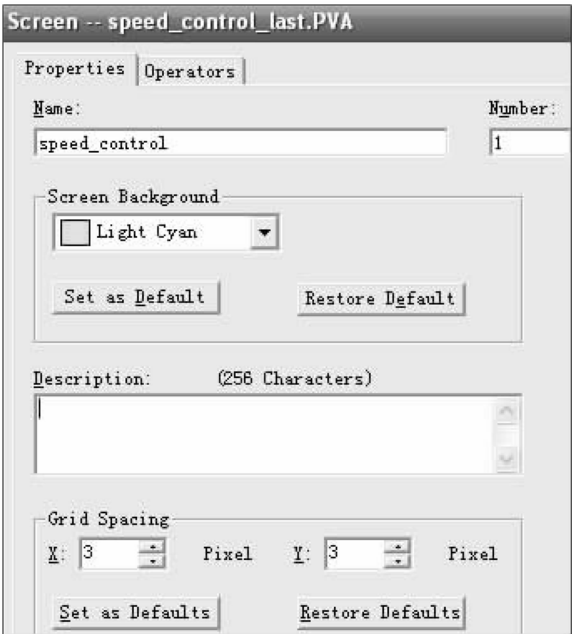


图 7-18　画面属性对话框

打开画面，首先编写变频器的启动、停止和两个速度给定按钮。选择如图 7-19 所示的按钮，这里按钮类型有四种，包括暂态按钮、保持型按钮、锁定按钮和多态按钮。由于这里需要的按钮是执行一次，不用一直保持，所以选择暂态按钮。按钮的默认形状为矩形，在菜单栏中选择 Format/Shape，可以改变按钮的形状，这里改为圆形。双击按钮打开按钮属性设置对话框，如图 7-20 所示。按

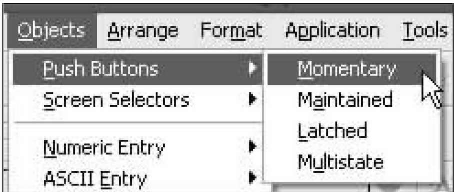


图 7-19　选择按钮

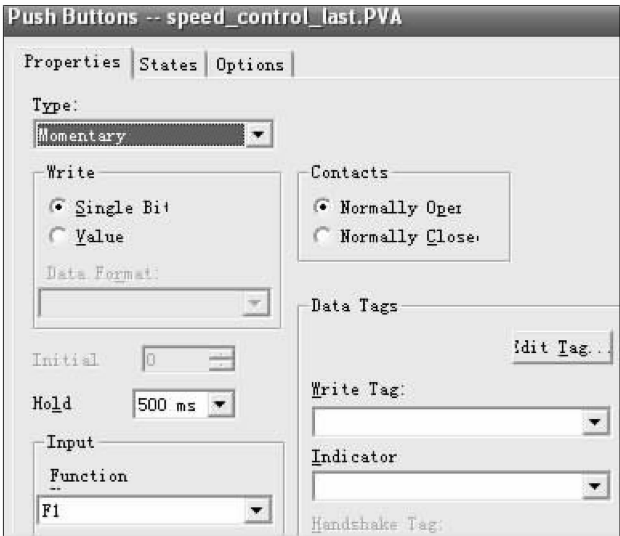


图 7-20　按钮属性设置对话框

钮类型选择暂态，Write 项中选择单个位，Contacts 选择常开，功能按钮选择 F1。然后点击编辑标签，弹出编辑标签的对话框，在标签名字处点击下拉选择，选择所需要的标签，如 start，则会弹出如图 7-21 所示的对话框，选择“是”即可，这时标签类型和地址都会自动出来。在按钮编辑对话框中的第二个按钮选项中，设置按钮的各种外观形象。



图 7-21　下载标签信息对话框

由于 PanelBuilder32 软件不支持中文，所以在写标注的时候可以用英文，也可以把中文注释做成图片粘贴到画面上。用同样的方法来编辑其他按钮。在编辑速度反馈、错误代码和速度给定值的显示的时候，选择如图 7-22 所示的数字显示器，在编辑变频器逻辑控制字和逻辑状态字的时候，选择如图 7-23 所示的指示器。



图 7-22　数字显示器

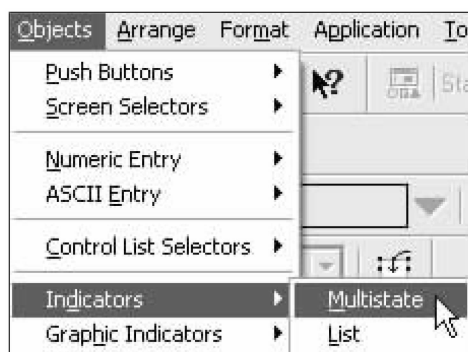


图 7-23　指示器

完成了此实验所需要的按钮和指示器以后，还要在画面上添加一个跳转到组态画面的按钮。如图 7-24 所示，选择跳转到组态界面按钮，并设置按钮的功能键为 F10。

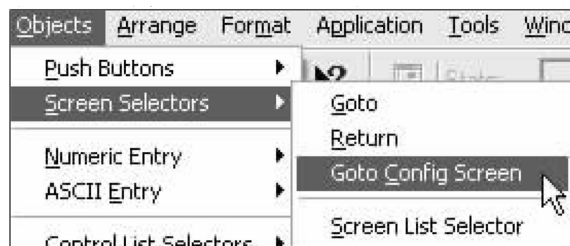


图 7-24　跳转到组态界面按钮

这样就完成了速度控制系统触摸屏画面的编写。

7.4 小结

本章主要讲述了以 Micro830 控制器为核心的速度控制系统的设计。首先讲述了速度控制系统的结构，并重点介绍了 PowerFlex4M 变频器的硬件特性和网络通信，然后介绍了速度控制系统的程序设计。通过本章的学习，可以更进一步的了解 Micro800 系列控制器的特性，以及具体的实用性。为 Micro800 系列控制器的应用起到很好的启蒙作用。

习　题

1. 已知 PowerFlex 4M 变频器中，地址为 8192 的寄存器是变频器接收控制器给的控制命令存放地址，请问 Micro 830 控制器通过 MSG _ MODBUS 指令给变频器写控制命令时，其地址位写 8192 是否正确？不正确该如何填写地址？

2. PowerFlex 4M 变频器中速度给定值寄存器地址是什么？状态逻辑字和速度反馈值的寄存器地址是什么？

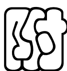
3. MSG _ MODBUS 指令需要哪些数据类型的数据？Micro 830 控制器用该指令通过通道 6 给 PowerFlex 4M 变频器一个 30Hz 的速度命令，该如何配置指令信息？

4. 使用 Micro 830 控制器如何实现电动机速度反馈给控制器使用（参见第 6 章 HSC 指令）？

思考题

1. PC 能否通过 Micro 830 控制器的串口访问组态的 2080-SERIALISOL 模块，进而通过 RS-485 通信口配置变频器参数？

2. 通过 MSG _ MODBUS 指令能否访问到变频器的电流反馈参数？如何实现？

Micro800 系列控制器见表 1。

表 1 Micro800 系列控制器

属 性	Micro810	Micro830				Micro850	
	12 点	10 点	16 点	24 点	48 点	24 点	48 点
本地通信端口	USB 2.0(带 USB 适配器)可使用任意标准 USB 打印机电缆	USB 2.0(非隔离型)(可使用任意标准 USB 打印机电缆) RS-232/RS-485 非隔离型复用串行端口				USB 2.0(非隔离型) RS-232/RS-485 非隔离型复用串行端口 10/100 Base T 以太网端口(RJ-45)	
本地串行端口协议	不支持	Modbus 主站/从站,ASCII/二进制,CIP 串口服务器					
本地数字量 I/O 点数	12	10	16	24	48	24	48
程序步数	2K(1 个程序步 = 12 个数据字节)	4K	4K	10K	10K	10K	10K
数据字节数	2KB	8KB	8KB	20KB	20KB	20KB	20KB
本地模拟量 I/O 通道数	可以将 4 个 DC24V 数字量输入配置为 0 ~ 10V 模拟量输入(仅限直流输入型)	通过功能性嵌入式模块提供				通过功能性嵌入式模块和扩展式模块提供	
预留嵌入式模块数插槽数量	0	2	2	3	5	3	5
最大允许的数字量 I/O 点数	12	26	32	48	88	132	
支持的附件或功能性嵌入式模块	● 液晶显示器,带有备份存储模块 ● USB 适配器	所有功能性嵌入式模块(参见 Micro830、Micro850 章节)					
支持的扩展式模块类型	不支持	不支持				所有扩展式模块	
电源	本地 120/240V 交流和 12/24V 直流选件	基本单元内置了 24V 直流电源,此外还提供可选的外部 120/240V 交流电源					
基本指令速度	每个基本指令为 2.5μs	每个基本指令为 0.30μs					
软件	Connected Components Workbench						

Micro800 系列控制器及附件功耗见表 2。

表 2　Micro800 系列控制器及附件功耗

控制器/模块	功　　耗	
Micro810 12 点(带或不带液晶显示屏)	3W(交流模块为 5VA)	
Micro830 和 Micro850(不带嵌入式/扩展式模块)	10/16 点	5W
	24 点	8W
	48 点	11W
嵌入式模块,每个	1.44W	
扩展 I/O(系统总线功率消耗)	2085-IQ16	-0.85W
	2085-IQ32T	-0.95W
	2085-IA8	-0.75W
	2085-IM8	-0.75W
	2085-OA8	-0.90W
	2085-OB16	-1.00W
	2085-OV16	-1.00W
	2085-OW8	-1.80W
	2085-OW16	-3.20W
	2085-IF4	-1.70W
	2085-IF8	-1.75W
	2085-OF4	-3.70W
	2085-IRT4	-2.00W

参 考 文 献

［1］ 钱晓龙，李鸿儒. 智能电器与 MicroLogix 控制器［M］. 北京：机械工业出版社，2003.

［2］ 钱晓龙. MicroLogix 控制器应用实例［M］. 北京：机械工业出版社，2003.

［3］ 钱晓龙，李晓理. 循序渐进 PowerFlex 变频器［M］. 北京：机械工业出版社，2007.

［4］ 钱晓龙，李晓理. 循序渐进 SL C500 控制系统与 PanelView 训练课［M］. 北京：机械工业出版社，2008.

［5］ 钱晓龙，路阳，袁伟. 循序渐进 Kinetix 集成运动控制系统［M］. 北京：机械工业出版社，2008.

Rockwell Automation

罗克韦尔自动化技术丛书

书　名	书　号	定 价
1、循序渐进Power Flex变频器	20376-6	33.00
2、循序渐进Kinetix集成运动控制系统	24498-1	45.00
3、电气控制与可编程序控制器应用技术	22095-4	49.00
4、ControlLogix系统实用手册	23037-3	58.00
5、循序渐进SLC500控制系统与PanelView训练课	23038-0	45.00
6、深入浅出NetLinx网络架构	24983-2	30.00
7、ControlLogix系统电力行业自动化应用培训教程	24127-0	58.00
8、循序渐进CMS机器控制系统	25797-4	47.00
9、ControlLogix系统水泥行业自动化应用培训教程	26077-6	39.00
10、MicroLogix核心控制系统	31237-6	59.00
11、ControlLogix系统在污水处理行业中的应用	35489-5	45.00
12、ControlLogix系统在给水处理行业中的应用	35490-1	45.00
13、PAC编程基本教程	36030-8	85.00
14、AB变频器及其控制技术	37125-0	34.00
15、Micro800控制系统	41257-1	34.00

地址：北京市百万庄大街22号
邮政编码：100037
电话服务
社服务中心：010-88361066
销售一部：010-68326294
销售二部：010-88379649
读者购书热线：010-88379203
网络服务
教材网：http://www.cmpedu.com
机工官网：http://www.cmpbook.com
机工官博：http://weibo.com/cmp1952
封面无防伪标均为盗版

上架指导：工业技术/电气自动化技术
ISBN 978-7-111-41257-1
策划编辑◎**林春泉** / 封面设计◎**鞠杨**

ISBN 978-7-111-41257-1



定价：34.00元